Sun Identity Manager 8.1 Upgrade

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

Part No: 820–5595
February 2009

090312@21990

Contents

Preface

Sun Identity Manager 8.1 Upgrade provides detailed information and instructions to help you upgrade your Sun^{TM} Identity Manager installation.

Note – If your current Identity Manager installation has a large amount of custom work, contact your Sun Professional Services representative about getting assistance with your upgrade.

Who Should Use This Book

This guide is for deployers and system administrators who are responsible for upgrading and configuring Sun Identity Manager 8.1 and associated software in a Production environment.

How This Book Is Organized

This guide is organized into the following chapters:

Chapter 1, "Overview of the Upgrade Process," provides general information about the upgrade process.

Chapter 2, "Planning," helps you identify goals and gather information to prepare for upgrading Identity Manager.

Chapter 3, "Developing and Testing Your Upgrade," provides guidelines for developing and verifying your upgrade procedure.

Chapter 4, "User Acceptance Testing," describes how to test your upgrade in an environment that simulates your Production environment.

Chapter 5, "Upgrading Your Production Environment," provides information and suggestions for upgrading your Production environment.

Chapter 6, "Skip-Level Upgrade Considerations," describes extra steps that you must perform to upgrade Identity Manager more than one version at a time.

Chapter 7, "Assessment Worksheets," provides worksheets that help you gather information to prepare for upgrading.

Related Books

The Sun Identity Manager 8.1 documentation set includes the following books.

Primary Audience	Title	Description
All Audiences	*Sun Identity Manager Overview*	Provides an overview of Identity Manager features and functionality. Provides product architecture information and describes how Identity Manager integrates with other Sun products, such as Sun Open SSO Enterprise and Sun Role Manager.
	Sun Identity Manager 8.1 Release Notes	Describes known issues, fixed issues, and late-breaking information not already provided in the Identity Manager documentation set.
System Administrators	*Installation Guide*	Describes how to install Identity Manager and optional components such as the Sun Identity Manager Gateway and PasswordSync.
	Upgrade Guide	Provides instructions on how to upgrade from an older version of Identity Manager to a newer version.
	System Administrator's Guide	Contains information and instructions to help system administrators manage, tune, and troubleshoot their Identity Manager installation.
Business Administrators	*Business Administrator's Guide*	Describes how to use Identity Manager provisioning and auditing features. Contains information about the user interfaces, user and account management, reporting, and more.

Primary Audience	Title	Description
System Integrators	*Deployment Guide*	Describes how to deploy Identity Manager in complex IT environments. Topics covered include working with identity attributes, data loading and synchronization, configuring user actions, applying custom branding, and so on.
	Deployment Reference	Contains information about workflows, forms, views, and rules, as well as the XPRESS language.
	Resources Reference	Provides information about installing, configuring, and using resource adapters.
	Service Provider 8.1 Deployment Guide	Describes how to deploy Sun Identity Manager Service Provider, and how views, forms, and resources differ from the standard Identity Manager product.
	Web Services Guide	Describes how to configure SPML support, which SPML features are supported (and why), and how to extend support in the field.

Documentation Updates

Corrections and updates to this and other Sun Identity Manager publications are posted to the Identity Manager Documentation Updates website:

`http://blogs.sun.com/idmdocupdates/`

An RSS feed reader can be used to periodically check the website and notify you when updates are available. To subscribe, download a feed reader and click a link under Feeds on the right side of the page. Starting with version 8.0, separate feeds are available for each major release.

Related Third-Party Web Site References

Third-party URLs are referenced in this document and provide additional, related information.

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Documentation, Support, and Training

The Sun web site provides information about the following additional resources:

- Documentation (`http://www.sun.com/documentation/`)
- Support (`http://www.sun.com/support/`)
- Training (`http://www.sun.com/training/`)

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. To share your comments, go to `http://docs.sun.com` and click Feedback.

Typographic Conventions

The following table describes the typographic conventions that are used in this book.

TABLE P–1 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`

`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`**
		`Password:`

aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.

TABLE P–1 Typographic Conventions *(Continued)*

Typeface	Meaning	Example
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX® system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE P–2 Shell Prompts

Shell	Prompt
C shell	`machine_name%`
C shell for superuser	`machine_name#`
Bourne shell and Korn shell	`$`
Bourne shell and Korn shell for superuser	`#`

Note – The Windows command-line prompt is `C:\`.

CHAPTER 1

Overview of the Upgrade Process

This chapter provides an overview of the Sun™ Identity Manager upgrade process and covers the following topics:

- "Why Upgrade?" on page 13
- "Upgrade Paths and Support Policies" on page 13
- "How Easy Is Upgrading?" on page 15
- "Terminology and Concepts" on page 16
- "Phases of the Upgrade Project" on page 19

Why Upgrade?

Upgrading your release of Sun Identity Manager software (Identity Manager) provides several advantages, including the following:

- Access to advanced features and functionality
- A more secure environment for your servers
- Continued eligibility for full support and services

Upgrade Paths and Support Policies

This section provides the following information:

- "Identity Manager Upgrade Paths" on page 14
- "End of Service Life for Software Support" on page 14
- "Identity Manager Deprecation Policy" on page 14

Identity Manager Upgrade Paths

To determine the upgrade path you must follow when upgrading to a newer version of Identity Manager, see "Upgrade Paths and Support Policies" in *Sun Identity Manager 8.1 Release Notes*. Generally, use the latest available patch or service pack for the version to which you are upgrading.

Using nonstandard upgrade techniques is *strongly* discouraged because serious, nonobvious consequences can result.

The Identity Manager standard upgrade process performs steps that are necessary to convert existing repository objects into the format that the newer version of Identity Manager expects. In particular, the *updater* programs that are shipped with each version of Identity Manager contain special logic that preserves the original meaning and behavior of these objects. Each updater program updates a specific type of configuration object, and each updater is invoked by an `ImportCommand` within `update.xml` or within a file that `update.xml` includes. Updated versions of preexisting objects often use slightly different mechanisms than those used in the sample objects that ship with the new Identity Manager version.

Caution – If you are upgrading Identity Manager, you must follow the required upgrade path noted in "Upgrade Paths and Support Policies" in *Sun Identity Manager 8.1 Release Notes*.

Even if you plan to perform a clean installation of Identity Manager and migrate your customizations to the new Identity Manager version, you must apply the standard upgrade process for each version of Identity Manager to properly update existing repository objects. The updater programs that are shipped with each version of Identity Manager work *only* with that version.

Simply comparing the objects that are produced by importing the new `init.xml` file with the objects that are produced by importing the old `init.xml` file is not a sufficient way to detect the changes that must be made in order to upgrade existing repository objects. Comparing these two sets of objects is a quick way to estimate the effort associated with an upgrade, but the standard upgrade path is the best way to achieve a safe and complete upgrade.

End of Service Life for Software Support

For details about the End of Service Life (EOSL) policy for Identity Manager software see "End of Service Life for Software Support" in *Sun Identity Manager 8.1 Release Notes*.

Identity Manager Deprecation Policy

For details about the deprecation policy for Identity Manager software see "Identity Manager Deprecation Policy" in *Sun Identity Manager 8.1 Release Notes*.

Note – Deprecations are announced in Chapter 6, "Deprecated APIs," in *Sun Identity Manager 8.1 Release Notes*.

When upgrading, always check the Release Notes for the new Identity Manager version to get the latest information about deprecated features.

How Easy Is Upgrading?

Basically, how easy or how difficult it is to upgrade Identity Manager depends on how much you customize Identity Manager and what kind of customizations you make. Some customizations are highly backward-compatible. For example, older workflows, forms, and rules tend to work without modification on newer versions of Identity Manager. Other customizations are more volatile and require additional work to integrate them with new versions of Identity Manager. For example, you might have to recompile or rewrite integration code that invokes Java classes that are part of the Identity Manager product.

The most significant effect of customization, however, is that customization changes how you must approach upgrading Identity Manager. Note that the following discussion uses terminology and concepts described in "Terminology and Concepts" on page 16.

Upgrading Identity Manager can be relatively simple if you use the product as shipped—that is, if you *configure* Identity Manager but do not *customize* it. If you are unfamiliar with these terms, see "Configurations and Customizations" on page 18. The upgrade process that is part of each version of the Identity Manager product was originally designed to be run in each environment. The product upgrade process automatically preserves your configuration settings as it updates the Identity Manager configuration objects and other repository objects to work with the newer version of code. The upgrade process, however, does not know how to update your customizations. In fact, the upgrade process ignores most customizations. The upgrade process does save off the customized version of any file that the upgrade process intends to replace, but this is the most that the upgrade process can do automatically.

If you customize Identity Manager, as the vast majority of customers do, then you cannot run the standard Identity Manager product upgrade in each environment. You must maintain and must retest your customizations after you upgrade the Identity Manager product, and you want to be certain that exactly what you tested goes into each environment. Because Identity Manager is often customized extensively, those who deploy Identity Manager have developed sophisticated practices for managing their customizations. For more information, see "Source Control and CBE" on page 17.

If you customize Identity Manager, the most common approach is to manage your customizations by maintaining a baseline and then to generate and promote an image of your Identity Manager application into each target environment. See "Baseline and Image" on page 17. You apply the standard Identity Manager product upgrade process only in your

Development environment. You then apply to your application baseline most of the changes that the upgrade makes. See "Identity Manager Product Compared With Identity Manager Application" on page 16. However, your upgrade procedure also must include some pieces of the Identity Manager product upgrade. For example, your upgrade procedure will include some subset of `update.xml` to update repository objects that are not managed as part of your baseline. Your upgrade procedure might also include database table scripts to update the Identity Manager repository table definitions.

Because the vast majority of customers do customize Identity Manager, and because many customize Identity Manager extensively, this document assumes that you have customizations and that you manage your customizations with this approach.

Terminology and Concepts

This section describes some specialized terminology that relates to upgrading Identity Manager.

Identity Manager Product Compared With Identity Manager Application

In some sections, this document distinguishes between the Identity Manager *product* and your Identity Manager *application*.

- Identity Manager product means the product as shipped, with standard samples, JSP files, JARs, and configuration objects.
- Identity Manager application means the customer's deployment of Identity Manager, which will be uniquely configured and might be customized, might include custom code and modified JARs, might be relabeled, and so forth.

Development, Test, QA, and Production Environments

This document assumes that you are using the following, different types of environments:

- A *Development environment* is where you configure, customize, and use source control to build an image of the Identity Manager application to be promoted to another environment. You also write an upgrade procedure in this environment that you follow in each target environment.
- A *Test environment* is where you test your upgrade procedure against controlled data and perform controlled testing of the resulting Identity Manager application.
- A *QA environment* is where you test your upgrade procedure against data, hardware, and software that closely simulate the Production environment and where you allow intended users to test the resulting Identity Manager application.

- A *Production environment* is where the Identity Manager application is actually available for business use.

Promotion

In this document, *promotion* refers to the process of generating a particular version of your Identity Manager application and moving that version from Development to Test, from Test to QA, and finally from QA to Production.

Source Control and CBE

You should use source-control tools to manage your configuration settings, any custom configuration objects, any custom code, any test plans, and any automated tests. Any source-control tool, such as CVS or Subversion, should allow you to track changes to any number of individual, text, or binary files and to reproduce a particular version of the Identity Manager application. This document refers to these tools generically as *source control*.

Some of these source-controlled files must be *tailored* for each environment. In particular, configuration objects contain values that often change for each environment. A particular Identity Manager resource object might point to one host machine in your Development environment, to another host machine in your Test environment, and to a third host machine in your Production environment.

Many customers use the Identity Manager IDE plug-ins for NetBeans™ and Eclipse to generate a tailored version of the Identity Manager application for each environment. The Identity Manager IDE plug-ins include a Configuration Build Environment (CBE). Some customers use older versions of the CBE that predate the Identity Manager IDE plug-in. A few customers use other tools and approaches, including proprietary or home-grown mechanisms. This document refers to these tools generically, and to the Identity Manager IDE specifically, as *CBE*.

The Identity Manager IDE plug-in and older CBE versions parameterize configuration objects (and can parameterize code) by replacing environment-specific values with placeholder values. To generate an image of the Identity Manager application that is appropriate to each environment, these tools replace the placeholder values with configured values that are appropriate to each environment. If you manage the parameterized objects in source control, you can “build out” the same version of the Identity Manager application for each target environment using a process that is predictable and repeatable.

Baseline and Image

In this document *baseline* refers to a tagged level of artifacts, which includes configuration objects, libraries, and custom code in source control, that corresponds to a particular version of your Identity Manager application. Your CBE can generate from this baseline an *image* of the Identity Manager application that is appropriate to each target environment. An image is a complete, working copy of the application that has been tailored for a specific environment.

At a minimum, your source control must contain a baseline for the version of your Identity Manager application that is currently deployed in your Production environment. Your source control can also contain baselines for previous versions of your Identity Manager application and baselines for versions of your Identity Manager application that you are still developing in preparation to roll out new functionality (such as modified configurations, modified workflows, modified forms, and new custom code for integrations).

Any baseline should be complete enough that you can use it to rebuild the corresponding version of your Identity Manager application. Some customers include the Identity Manager product itself within that baseline. Other customers save the Identity Manager product image separately and use the baseline artifacts to overlay that product image with their own configurations and customizations.

Skip-Level Upgrade

A *skip-level*, or *multi-hop*, upgrade updates your Production environment to an Identity Manager product version that is beyond the next major release from its current version. A skip-level upgrade typically requires a series of upgrades, but it is technically possible, and some customers find it feasible, to develop a single upgrade procedure that updates a target environment directly to the target Identity Manager version.

Note – Chapter 6, "Skip-Level Upgrade Considerations," describes special considerations for a multi-hop upgrade. If you are uncertain about how to proceed, or if these considerations seem unclear to you, contact Sun Professional Services for assistance with planning and executing your upgrade.

Configurations and Customizations

In this document, the term *configuration* means editing configuration objects in the Identity Manager product. You can edit configuration objects by using the Identity Manager Administrator Interface or by editing repository objects in accordance with published documentation. For example, configuration includes specifying values in System Configuration or in a Reconciliation Policy. Configuration also includes defining Resource objects that represent the target systems and applications that Identity Manager will manage.

For the purposes of this document, the term *customization* refers to any code, file, or repository object that you write. Customizations include adding your own Java code, adding or modifying JSP files, and writing your own XPRESS code such as workflow, forms, and rules. Creating your own configuration objects or message catalogs is also considered customization.

Note that Sun Professional Services defines these terms somewhat differently. For example, Sun Professional Services might consider customer-specific workflows to be *configurations*.

In this document, however, the key distinction is that the Identity Manager product upgrade automatically preserves and updates customer-specified values within certain well-known objects and within instances of certain well-known types of objects. These are *configurations*. The Identity Manager product upgrade does not attempt to modify customer-written code or objects. These are *customizations*.

Upgrade Process and Upgrade Procedure

In this document, *upgrade process* refers to the upgrade mechanisms that are part of every version of the Identity Manager product. Each new version of Identity Manager contains not only updated code, but also an `update.xml` script. This `update.xml` script carefully preserves your configuration settings as it updates existing repository objects to work with the updated code. Any full release of Identity Manager might also contain sample database table scripts. These sample database table scripts carefully migrate existing data and update your database table definitions to work with the updated code.

This document uses the term *upgrade procedure* to refer to a document that you develop and maintain. An upgrade procedure often takes the form of a checklist. Your upgrade procedure states exactly what you plan to do in each target environment to upgrade your Identity Manager application. See "Task 4: Prepare Your Upgrade Procedure" on page 31.

Upgrade Project

This document describes phases in an *upgrade project*. An upgrade project is your effort to upgrade your Identity Manager application to work with a new version of the Identity Manager product on which your application is based. An upgrade project includes your efforts to maintain and retest your configurations and customizations after executing the Identity Manager upgrade process in your Development environment. An upgrade project also includes your efforts to maintain and retest the upgrade procedure that you will follow in each target environment.

Phases of the Upgrade Project

The following figure shows the major phases of the upgrade project, summarizing the tasks that you must perform to complete each phase. The rest of this document guides you through each phase.

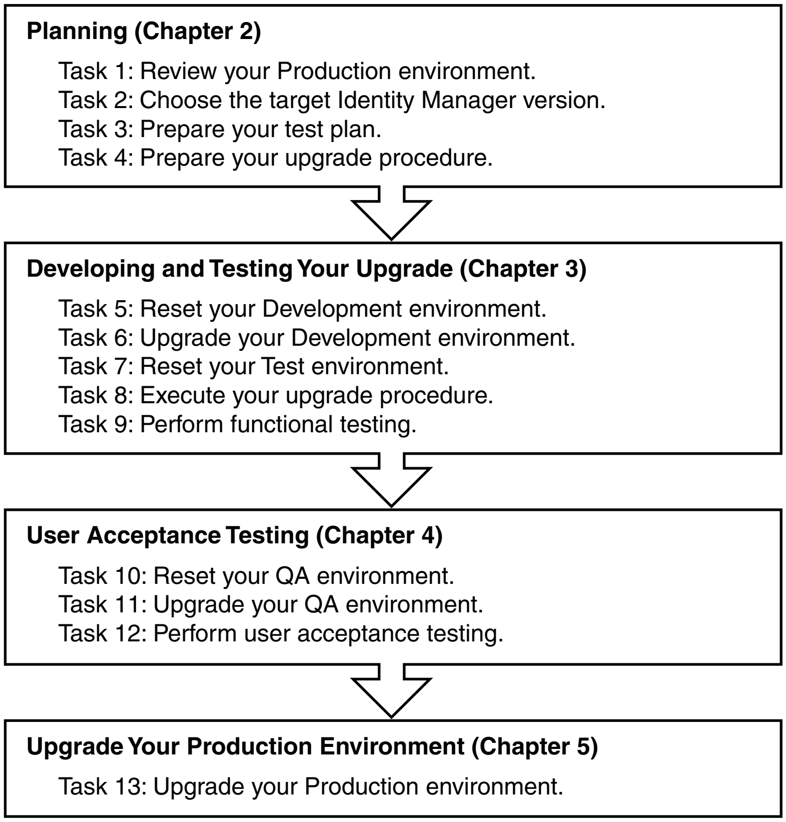

Planning (Chapter 2)
Task 1: Review your Production environment.
Task 2: Choose the target Identity Manager version.
Task 3: Prepare your test plan.
Task 4: Prepare your upgrade procedure.
Developing and Testing Your Upgrade (Chapter 3)
Task 5: Reset your Development environment.
Task 6: Upgrade your Development environment.
Task 7: Reset your Test environment.
Task 8: Execute your upgrade procedure.
Task 9: Perform functional testing.
User Acceptance Testing (Chapter 4)
Task 10: Reset your QA environment.
Task 11: Upgrade your QA environment.
Task 12: Perform user acceptance testing.
Upgrade Your Production Environment (Chapter 5)
Task 13: Upgrade your Production environment.

FIGURE 1–1 Upgrade Phases

CHAPTER 2

Planning

Careful preparation allows for a smoother upgrade. Listing your goals for the upgrade can help you make decisions that are appropriate for your company's needs.

The Planning phase of the upgrade process includes the following tasks:

- "Task 1: Review Your Production Environment" on page 21
- "Task 2: Choose the Target Identity Manager Version" on page 28
- "Task 3: Prepare Your Test Plan" on page 30
- "Task 4: Prepare Your Upgrade Procedure" on page 31

Task 1: Review Your Production Environment

Upgrading to a newer Identity Manager release might require changes to the platform in your environment. You can determine the best upgrade path and estimate the complexity of the upgrade by assessing and documenting your Production environment.

This section describes the steps that you perform when reviewing your Production environment:

- "Step 1: Document Your Platform" on page 22
- "Step 2: Document Your Identity Manager Installation" on page 23
- "Step 3: Document Your Custom Components" on page 25

Tip – If you use source control and CBE to manage this information, these can serve as the documentation for your Identity Manager installation and for your custom components. Review the information and familiarize yourself with the various environments in which you deploy your Identity Manager application, giving particular attention to your Production environment.

Step 1: Document Your Platform

To determine the best upgrade path, use the worksheets provided in Chapter 7, "Assessment Worksheets," to inventory the components of your current platform, including the following:

- "Application Servers" on page 22
- "Database Servers" on page 22
- "Identity Manager Gateway" on page 22
- "Java Runtime Environment" on page 23
- "Supported Resources" on page 23
- "Web Servers" on page 23

Note – Verify that you are using the correct version of these components for the upgrade version that you want to install. Check "Supported Software and Environments" in *Sun Identity Manager 8.1 Release Notes* for details.

Caution – If you are using an Oracle® repository, the Identity Manager repository DDL uses data types that are not properly handled by older Oracle JDBC drivers. The JDBC drivers in `ojdbc14.jar` do not properly read all of the columns in the log table.

You must upgrade to the `ojdbc5.jar` for JDK 5 drivers for Identity Manager to work properly.

Application Servers

Record the application server version, and note any additional patches or service packs. In addition, record the following:

- Operating system version and any additional patches or service packs.
- Java Development Kit (JDK™) version required by your application server. When upgrading Identity Manager, you must use a JDK supplied by the same vendor.

Database Servers

Record the database server version, and note any additional patches or service packs.

Identity Manager Gateway

Verify which Identity Manager Gateway version you are running by performing the following steps:

1. **Open a command window and execute the following command on each of the Gateway servers:**

```
gateway -v
```

2. **Record the results.**
3. **Record the operating system version of each Gateway server.**

Note – The Gateway server version should always be the same as the Identity Manager version.

Java Runtime Environment

Record the currently installed JRE™ version required by the `lh` console. Also record the name of the vendor that supplied the installed JRE (for example, Sun, IBM, Oracle, and so on). When upgrading Identity Manager, you must use a JRE supplied by the same vendor.

Supported Resources

Record supported resource names and versions, and note any additional patches or service packs.

Web Servers

Record the Web server version, and note any additional patches or service packs.

Step 2: Document Your Identity Manager Installation

To determine the best upgrade path, use the worksheets provided in Chapter 7, "Assessment Worksheets," to inventory the components of your current Identity Manager installation.

The following sections describe methods for collecting this information:

- "Identity Manager Version" on page 23
- "Identity Manager Assessment Tools" on page 24

Identity Manager Version

To verify the version number of your current Identity Manager installation, use the Identity Manager Console .

1. **From the command line, type:**

   ```
   lh console
   ```

2. **To display the Identity Manager version number, type:**

   ```
   version
   ```

Identity Manager Assessment Tools

Identity Manager provides the following utilities to list and record your installation information:

- `installed` **Utility**: Searches the `$WSHOME/bin` directory for manifests and provides version information for releases, patches, service packs, and hotfixes.
- `inventory` **Utility**: Inspects the file system for files that were added to or deleted from the system, using files that are packaged in the release. This utility determines which files were changed based on the manifest that shipped with Identity Manager.

To access the `installed` and `inventory` utilities, follow these steps:

1. **Open a command window and change directories to `$WSHOME/bin`.**
2. **At the prompt, execute the following command:**

   ```
   ./lh assessment
   ```
3. **At the prompt, type one of the following commands:**
   - **`installed`** [*option*] [*option*]**...**
   - **`inventory`** [*option*] [*option*]**...**

The following tables describe the options that you can use with the `installed` and `inventory` utilities.

`installed` **Utility Options**

Option	Function	Description
`-h`	Help	Displays usage.
`-r`	Releases	Displays only installed releases.
`-p`	Patches	Displays only installed patches.
`-s`	Service packs	Displays only installed service packs.
`-f`	Hotfixes	Displays only installed hotfixes.

Note – Be sure to record the manifest file names that are associated with all service packs or patches. For example:

```
Identity_Manager_8_0_0_0_20080530.manifest
```

`inventory` **Utility Options**

Option	Function	Description
`-a`	Added	Displays only added files.
`-d`	Deleted	Displays only deleted files.
`-h`	Help	Displays usage.
`-m`	Modified	Displays only modified files.
`-u`	Unchanged	Displays only unchanged files.

Step 3: Document Your Custom Components

Use the worksheets provided in Chapter 7, "Assessment Worksheets," to inventory your custom components, including the following:

- "Customized Database Table Definitions" on page 25
- "Custom File System Objects" on page 25
- "Custom Repository Objects" on page 26

Note –

- If you are using the Identity Manager IDE or an older version Consolidated Build Environment (CBE), these component customizations should already be part of your baseline. In this case, the CBE baseline serves as your documentation.
- If your current Identity Manager installation has a large amount of custom work, contact Sun Professional Services for assistance with your upgrade.

Customized Database Table Definitions

Version 7.1 and version 8.0 of Identity Manager made significant changes to the Identity Manager database table definitions.

If you previously modified the database table definitions for the Identity Manager repository, you must decide whether to make the same modifications to the new and updated tables.

Custom File System Objects

You might need to update your customized file system objects to enable them to function properly with later Identity Manager releases. List any customized file system object names that are in your environment as explained in the following sections.

Modified JavaServer Pages Files

Recent Identity Manager versions might contain API changes. If you have modified `.jsp` files in your installation, you might have to update them when upgrading. You must update any JSP that was supplied by Identity Manager and changed during a deployment (or a custom JSP that uses Identity Manager APIs) to work with the new JSP structure and API changes for the target release.

Note – For a detailed description of API changes, see the Identity Manager Release Notes for the release to which you are upgrading.

Use the `inventory -m` command (described on "Identity Manager Assessment Tools" on page 24) to identify any JSP modifications made in your deployment.

For more information about JSP customizations, see Chapter 11, "Editing Configuration Objects," in *Sun Identity Manager 8.1 Technical Deployment Overview*.

Modified `Waveset.properties` File

Record any changes that you made to the default `Waveset.properties` file.

Modified `WPMessages.properties` File

Record any changes that you made to the default `WPMessages.properties` file.

Customized Property Files

Record any changes that you made to other property files on your system.

Custom Resource Adapters (and Other Custom Java)

You might have to recompile your custom resource adapters, depending on the target Identity Manager version. All custom Java code that uses Identity Manager APIs (including custom resource adapters) requires a recompile during upgrading. Also, consider other Java classes that use the Identity Manager library.

Modified Stylesheets

Record any changes that you made to the Identity Manager stylesheets.

Custom Repository Objects

You might have to maintain customized repository objects to enable them to function properly with target Identity Manager releases. Record any customized repository objects that are in your environment as explained in the following sections.

Note – You can use the Identity Manager SnapShot feature to create a baseline or *snapshot* of the customized repository objects in your deployment, which can be very useful when planning an upgrade. See “Step 5: Take a Snapshot” on page 35 for more information.

Modified Forms

You might have to update customized forms to take advantage of current product enhancements.

Modified Workflows

You might have to update customized workflows to take advantage of current product enhancements.

Modified Email Templates

You might have to export customized email templates to take advantage of current product enhancements.

Custom Repository Schema

Significant schema changes occurred between Version 7.0 and Version 8.0 of Identity Manager. If you are upgrading from an earlier version of Identity Manager, you must update your schema.

Other Custom Repository Objects

Record the names of any other custom repository objects that you created or updated. You might have to export these objects from your current installation and then reimport them to the newer version of Identity Manager after upgrading.

Admin group
Admin role
Configuration
Policy
Provisioning task
Remedy configuration
Resource action
Resource form
Role
Rule
Task definition
Task template
User form

Note – The SPML 2.0 implementation in Identity Manager changed in version 8.0. In previous releases, the SPML `objectclass` attribute used in SPML messages was mapped directly to the `objectclass` attribute of Identity Manager `User` objects. The `objectclass` attribute is now mapped internally to the `spml2ObjectClass` attribute and is used internally for other purposes.

During the upgrade process, the `objectclass` attribute value is automatically renamed for existing users. If your SPML 2.0 configuration contains forms that reference the `objectclass` attribute, you must manually change those references to `spml2ObjectClass`.

Identity Manager does not replace the sample `spml2.xml` configuration file during an upgrade. If you used the `spml2.xml` configuration file as a starting point, be aware that this file contains a form with references to `objectclass` that you must change to `spml2ObjectClass`. Change the `objectclass` attribute in forms (where it is used internally), but *do not* change the `objectclass` attribute in the target schema (where the attribute is exposed externally).

You can use the Identity Manager SnapShot feature to copy the following, specific object types from your system for comparison:

AdminGroup	ResourceAction
AdminRole	Resourceform
Configuration	Role
EmailTemplate	Rule
Policy	TaskDefinition
ProvisionTask	TaskTemplate
RemedyConfig	UserForm

For specific instructions, see "Step 5: Take a Snapshot" on page 35.

Task 2: Choose the Target Identity Manager Version

Note – For the most current description of Identity Manager upgrade paths, see the "Upgrade Paths and Support Policies" in *Sun Identity Manager 8.1 Release Notes*.

In general, you should upgrade to the most recent Identity Manager release that is available during your testing time frame. For example, assume that you are testing now with version 7.1.1, as this Identity Manager version was the most current release available when you started your current test cycle. Assume further that the next new release, 7.1.2, is scheduled for July 10th, and that July 15th is the projected start date of your next test cycle. You should plan to upgrade to 7.1.2 when you start your next test cycle.

Be sure that the platform in your Production environment supports the new version of the Identity Manager product. If not, plan to update the platform in each environment before you upgrade your Identity Manager application. Reset each target environment to match the Production platform before upgrading that target environment. In general, you must update your platform as part of the upgrade procedure that you follow in each target environment.

In cases where both your current Identity Manager product version and the target Identity Manager version support the updated platform, you can update your platform as a separate change and promote this change all the way to your Production environment before upgrading your Identity Manager application.

The standard upgrade processes that are part of each full release of Identity Manager generally upgrade an existing installation from any version of the previous major release.

Review the Release Notes for the target version of Identity Manager to which you plan to upgrade. The Release Notes document release-specific upgrade considerations. They also contain documentation addenda, bug fixes, and known issues.

Consider your configurations and customizations, and then identify any changes in the Identity Manager product that might affect those configurations and customizations.

Check your current release to see which hotfixes you have installed. Find the bug number associated with each hotfix, and check the Release Notes to confirm that the new, target Identity Manager version contains all of the hotfixes you need.

Note – Sun's new *patch process* replaces the older hotfix process. The patch process is cumulative, so you can expect fewer problems with unique fixes. The patch process also makes it easier for you to track a fix by its actual bug number. However, it is still possible that a fix made against an older version might not yet be available in a newer version. Regardless of which process your current version of Identity Manager follows, you must confirm that the new, target Identity Manager version contains all of the bug fixes that you need.

Note – If you want to upgrade your Identity Manager application more than one level (that is, beyond the next major version from your current version), you must read "Phases of a Skip-Level Upgrade" on page 66, which describes how a skip-level upgrade changes the tasks described in this section.

Task 3: Prepare Your Test Plan

Before proceeding to the next phase of the upgrade, be sure that you have prepared a current, comprehensive test plan. The goal of a test plan is to confirm that all your current Identity Manager application functionality remains intact through the upgrade process.

- If you have an existing test plan, read "Review Your Existing Test Plan" on page 30.
- If you have not a prepared test plan, create one now using the guidelines described in "Create a Test Plan" on page 30.

Review Your Existing Test Plan

Does your existing test plan address everything that you want to test? Is it up-to-date? Is it specific? If not, you must revise your test plan appropriately.

If you are particularly concerned with the performance of a particular set of functions or with items such as the amount of system memory or database space the Identity Manager application consumes, then be sure that your test plan also measures these items.

After upgrading the Identity Manager product or after making any significant change to your Identity Manager configurations or customizations, be sure to retest your Identity Manager application.

Create a Test Plan

You must create a test plan if you do not already have one prepared for your Identity Manager application. A generic test plan includes:

1. Introduction
   - Description of this document
   - Related documents
   - Schedule and milestones
2. Resource requirements
   - Hardware
   - Software (test tools)
   - Staffing
3. Features you are going to test and the test approach
   - New features testing
   - Regression testing
4. Features you are not going to test
5. Test deliverables

6. Dependencies and risks
7. Entrance and exit criteria

Task 4: Prepare Your Upgrade Procedure

Before proceeding to the next phase of the upgrade, be sure that you have prepared a current, comprehensive upgrade procedure. See "Upgrade Process and Upgrade Procedure" on page 19 for more information.

The goal of an upgrade procedure is to specify exactly who does what as you upgrade your Identity Manager application in each environment. You will develop and maintain this upgrade procedure as you upgrade your Identity Manager application in each environment.

- If you have an existing upgrade procedure, read "Review Your Existing Upgrade Procedure" on page 31.
- If you have not prepared an upgrade procedure, create one now using the guidelines described in "Create an Upgrade Procedure" on page 31.

Review Your Existing Upgrade Procedure

Does your existing upgrade procedure specify exactly who does what and when as you upgrade your Identity Manager application in each environment? Is it clear how and why the procedure differs in each environment? Is your procedure up-to-date? Does your upgrade procedure contain the same steps for your Test environment and for your QA environment that it does for your Production environment? If not, you must revise your upgrade procedure appropriately.

Are there important considerations that are unique to your Production environment? If so, your upgrade procedure must rehearse the same steps in your QA environment. See "Special Considerations for Production" on page 63. If the duration of the upgrade procedure in your Production environment is important, then be sure that your upgrade procedure says to record the duration of each step in each environment. Upgrading your QA environment should give you a particularly good indication of how long it will take to upgrade your Production environment.

Create an Upgrade Procedure

You must create an upgrade procedure if you have not already prepared one for your Identity Manager application.

The following is generally true of an update procedure:

- Takes the form of a checklist.

Your upgrade procedure may include supporting documentation, but the administrator who performs the upgrade procedure will want a clear, complete, and concise set of instructions.

- Includes most, if not all, of the steps described in "Task 8: Execute Your Upgrade Procedure" on page 51.

  Your upgrade procedure is generally far more specific, spelling out exactly who must do what in each environment. For example, your procedure must include specific commands and specific parameter values that an administrator must issue in each environment.
- Includes additional steps.

  For example, you might have to stop and restart external processes if your Identity Manager application integrates with external applications. You might also be required to notify users or systems personnel before taking the Identity Manager application or other affected applications offline.
- Is the same for each target environment.

  Specific parameter values, such as host names and connection information, might vary from environment to environment. The steps in the procedure, however, should be the same in each environment. Even if, for example, there is no one to notify about application downtime in a Test environment or a QA environment, you should rehearse this step in each environment.
- Includes a timetable.

  Estimate the expected duration for each step, and record the actual duration of each step. The durations that you see in your QA environment are particularly important for predicting the durations that you will see in your Production environment.

◆ ◆ ◆ CHAPTER 3

Developing and Testing Your Upgrade

The developing and testing phase of the upgrade consists of the following tasks:

- "Task 5: Reset Your Development Environment" on page 33
- "Task 6: Upgrade Your Development Environment" on page 34
- "Task 7: Reset Your Test Environment" on page 50
- "Task 8: Execute Your Upgrade Procedure" on page 51
- "Task 9: Perform Functional Testing" on page 56

Task 5: Reset Your Development Environment

You must revert your existing Development environment, or create and set a new Development environment, to the Identity Manager application baseline that corresponds to your Production environment. You must also reset the platform in your Development environment to match the platform in your Production environment. For more information, see "Step 1: Document Your Platform" on page 22.

Use source-control tools to manage your configuration settings, any custom configuration objects, any custom code, test plans, and automated tests. For more information, see "Source Control and CBE" on page 17.

If your site's processes allow administrators to change Identity Manager configurations and customizations directly in the Production environment (for example, without updating the baseline version in source control), then you must compare the current production configurations and customizations to those in the source-control baseline. Identify any changes in the Production environment, apply each change to the Development environment, and retest as appropriate. Merge these changes into the source-control baseline for your Identity Manager application. If the production changes seem significant and cannot be fully tested in the Development environment, consider promoting the updated Identity Manager baseline to the Test environment and retesting that baseline before proceeding with the Identity Manager upgrade.

Task 6: Upgrade Your Development Environment

You must perform each of the following steps in your Development environment:

- “Step 1: Stop Active Sync and Reconciliation” on page 34
- “Step 2: Stop the Identity Manager Application” on page 34
- “Step 3: Back Up Your Identity Manager Application” on page 35
- “Step 4: Remove Hotfixes” on page 35
- “Step 5: Take a Snapshot” on page 35
- “To Take a Snapshot” on page 35
- “Step 6: Update Your Platform” on page 36
- “Step 7: Upgrade the Identity Manager Product” on page 37
- “Step 8: Take Another Snapshot” on page 45
- “Step 9: Analyze the Changes” on page 45
- “Step 10: Rebuild All Custom Java Classes” on page 48
- “Step 11: Make Necessary Changes in XPRESS” on page 48
- “Step 12: Test Your Identity Manager Application in the Development Environment” on page 48
- “Step 13: Restart Active Sync and Reconciliation” on page 49
- “Step 14: Merge Changes Back Into Source Control” on page 49

Note – You must perform some of these steps when upgrading any environment. However, many of these steps are unique to the Development environment because this is the environment where you update the baseline for your Identity Manager application.

Step 1: Stop Active Sync and Reconciliation

Set any Active Sync processes to start manually and, if applicable, disable any scheduled reconciliations until the upgrade is finished and appears to be successful.

Tip – Step 1 is optional, but performing this step is considered a best practice when upgrading the Production environment.

Also, if you perform Step 1 in your Production environment, make it a standard step when upgrading all of your other environments.

Step 2: Stop the Identity Manager Application

Quiesce your Identity Manager application and make it unavailable to all administrators and end users.

Step 3: Back Up Your Identity Manager Application

Make a copy of your existing database and Identity Manager file structure.

Backing up the database and file structure enables you to reinstate your working environment, if necessary.

Step 4: Remove Hotfixes

Remove any hotfix class files from your `WEB-INF/classes` directory.

Generally, a hotfix class file works only with the specific version of the Identity Manager product for which that hotfix was delivered.

Step 5: Take a Snapshot

Make a copy of your existing configuration objects. Also, make a copy of other types of objects in the repository or copy at least a representative sampling of those objects.

The Identity Manager product upgrade saves the file system artifacts that it overlays, such as JSP files, but the upgrade does not preserve “before-images” of every object that it modifies in the repository. Taking a snapshot enables you to detect changes that the Identity Manager product upgrade makes to objects in the repository.

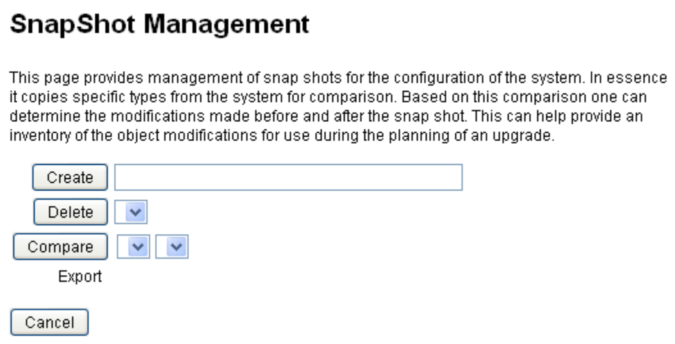
▼ To Take a Snapshot

The following instructions describe how to use the Identity Manager SnapShot feature to create a baseline of the customized repository objects in your deployment and how to compare two snapshots to determine what changes have been made to certain system objects before and after the upgrade.

Note – The SnapShot feature is not intended for detailed, ongoing XML diffs. It is only a minimal tool for “first-pass” comparisons.

---

1 **From the Identity Manager Debug page, click the SnapShot button to view the SnapShot Management page.**



SnapShot Management

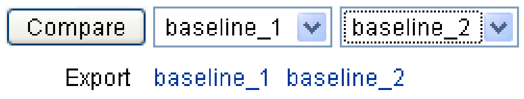
This page provides management of snap shots for the configuration of the system. In essence it copies specific types from the system for comparison. Based on this comparison one can determine the modifications made before and after the snap shot. This can help provide an inventory of the object modifications for use during the planning of an upgrade.

2 **Type a name for the snapshot in the Create field, and click the Create button.**

When Identity Manager adds the snapshot, the snapshot's name displays in the Compare menu list and to the right of the Export label.

3 **To compare two snapshots, do the following:**

a. **Select the snapshots from each of the two Compare menus:**



b. **Click the Compare button.**

- If no objects were changed, a message indicates that no differences were found.
- If object changes are detected, a message displays the object type and name, and indicates whether the object is different, absent, or present.

  For example, if an object is present in baseline_1, but not present in baseline_2, then the baseline_1 column indicates `Present` and the baseline_2 column indicates `Absent`.

4 **If you want to export a snapshot to a file in XML format, click the snapshot name link.**

5 **If you want to delete a snapshot, choose the snapshot name from the Delete menu and then click Delete.**

Step 6: Update Your Platform

If the target Identity Manager product version requires changes to your platform, you must make these changes before upgrading the Identity Manager product.

If you plan on upgrading the JDK or JRE, you must use a JDK or JRE supplied by the same vendor as your previous JDK. For example, do not install a Sun JDK if previously you were using a JDK from IBM.

Caution – If you are using an Oracle repository, the Identity Manager repository DDL uses data types that are not properly handled by older Oracle JDBC drivers. The JDBC drivers in `ojdbc14.jar` do not properly read all of the columns in the log table.

You must upgrade to the `ojdbc5.jar` for JDK 5 drivers for Identity Manager to work properly.

Step 7: Upgrade the Identity Manager Product

To upgrade the Identity Manager product itself, you might be required to do the following:

- "Update the Repository Database Tables" on page 37
- "Upgrade the Identity Manager Product" on page 38
- "Upgrade All Gateway Instances" on page 44
- "Upgrade All PasswordSync Instances" on page 45

Update the Repository Database Tables

Most major releases and some minor releases of Identity Manager include database table changes. Consequently, you might have to modify the sample SQL scripts for your environment.

You must also update the database tables if you made any of the following modifications:

- Changed the database instance name
- Changed the name of the database account that owns the database tables
- Separated the owner of the database tables from the database account used to connect to the database
- Made more advanced DBA changes, such as configuring specific table spaces and growth characteristics for different sets of tables and indexes

You must remember any changes that you make to the sample SQL scripts for each Identity Manager version and use source control to manage these changes. In the future, you will have to make similar changes to the sample SQL scripts for subsequent Identity Manager versions.

Upgrade the Identity Manager Product

You can use either of the following methods to upgrade the Identity Manager product:

- Use the Identity Manager installer program as described in "To Use the Identity Manager Installer" on page 39.
- Use the Identity Manager manual upgrade process as described in "Upgrading Manually" on page 40.

Both methods produce the same results.

Note – In some environments you might prefer using the manual upgrade procedure. For example:

- If you want to fully automate the upgrade as part of a repeatable upgrade procedure
- If you have restricted access to your Production environment or cannot start the console

Upgrading the Identity Manager product might modify objects in the Identity Manager repository and in some file system artifacts such as `.jsp` files, Identity Manager product JARs, and third-party JARs.

When upgrading the Identity Manager product, be aware of the following:

- If you copy files from the installation media to your own location, you must put the `idm.war` and `install.class` files in the same directory.
- Use only one Identity Manager server to import `update.xml`, and have only one Identity Manager server running during the upgrade.

  If you start any other Identity Manager servers during the upgrade, you must stop and restart those servers before making them available.
- If your application server is installed on a machine running a UNIX® system, change directories to the `$WSHOME/bin` directory and run the following command to allow the scripts in this directory to be executed:

  ```
  chmod -R +x *
  ```
- For UNIX environments, be sure that you have an `install` directory in one of the following locations and that you can write to that directory:

For Linux/HP-UX	`/var/opt/sun/install`
For Solaris™	`/var/sadm/install`

  Previously installed hotfixes are archived in the `$WSHOME/patches/HotfixName` directory.
- The upgrade program has three steps: the upgrade pre-process step, the upgrade step, and the upgrade post-process step. The upgrade post-process step runs in a separate Java virtual machine and the default heap size for this step is 1024 MB. If you experience out-of-memory exceptions during an upgrade, set this value higher. To specify a custom value, set the `JAVA_OPTS` environment variable using the form –`Xmx`<*heap size*> where heap size is a value, such as `2048m`. An example is `-Xmx2048m`.

▼ To Use the Identity Manager Installer

Use the Identity Manager installation and upgrade program to upgrade your Development environment.

1 **Use one of the following methods to start the installer:**

- To use the GUI installer, run `install.bat` (for Windows) or `install` (for UNIX).

  The installer displays the Welcome screen.
- To activate the installer in `nodisplay` mode, change to the directory where the software is located, and type:

  ```
  install -nodisplay
  ```

  The installer displays the Welcome text, and then presents a list of questions to gather installation information in the same order as the GUI installer.

  If no display is present, the installer defaults to the `nodisplay` option.

  The installer does not install an older version of the software over a newer version. In this situation, an error message displays and the installer exits.

2 **On the Welcome screen, click Next.**

3 **On the Install or Upgrade? screen, select Upgrade and click Next.**

4 **On the Select Installation Directory screen, select the directory where the earlier Identity Manager version is located and click Next.**

The installer displays progress bars for the pre-upgrade and post-upgrade processes and then proceeds to Installation Summary screen.

5 **For detailed information about the installation, click Details, view the log file, and click Close to exit the installer.**

6 **Remove all of the compiled Identity Manager files from the work directory of the application server.**

Upgrading Manually

In some environments, you might want to perform the upgrade steps manually instead of using the Identity Manager installation and upgrade program.

- Make sure that you set the `JAVA_HOME` environment variable.
- Make sure that the `bin` directory in the `JAVA_HOME` directory is in your path.
- Any previously installed hotfixes will be archived to the `$WSHOME/patches/`*HotfixName* directory.

Note – The instructions in this section are based on installing Identity Manager on a Tomcat application server. Depending on your application server, you might have to use slightly different commands.

Refer to the appropriate chapter in Part II, "Installing Identity Manager," in *Sun Identity Manager 8.1 Installation* for application server-specific instructions.

▼ To Perform a Manual Upgrade on a Windows Platform

Perform the following steps to upgrade Identity Manager manually on a supported Windows platform:

1. **Stop the application server and Gateway.**

2. **Update the Identity Manager database.**

3. **Enter the following commands to set your environment:**

```
set ISPATH=Path-to-install-software
set WSHOME=Path-to-Identity-Manager-Installation  OR Staging-Directory  set TEMP=Path-to-Temporary-Directory
```

Note – If you have a space in the path to the Identity Manager installation directory, you must specify the `WSHOME` environment variable without double quotes ("), as shown in the following example.

Do not use trailing slashes (\) when specifying the path even if the path contains no spaces.

```
set WSHOME=c:\Program Files\Apache Group\Tomcat 6.0\idm
```

or

```
set WSHOME=c:\Progra~1\Apache~1\Tomcat~1\idm
```

The following path will not work:

```
set WSHOME="c:\Program Files\Apache Group\Tomcat 6.0\idm"
```

---

4. **Run the pre-process.**

```
mkdir %TEMP%
cd /d %TEMP%
jar -xvf %ISPATH%\IDM.WAR\
WEB-INF\lib\idm.jar WEB-INF\lib\idmcommon.jar
set TMPLIBPTH=%TEMP%\WEB-INF\lib
set CLASSPATH=%TMPLIBPTH%\idm.jar;\
%TMPLIBPTH%\idmcommon.jar;
java -classpath %CLASSPATH% -Dwaveset.home=%WSHOME%\
   com.waveset.install.UpgradePreProcess
```

5 **Install the software.**

```
cd %WSHOME%
jar -xvf %ISPATH%\IDM.WAR
```

6 **Run the post-process.**

```
java -classpath %CLASSPATH% -Dwaveset.home=%WSHOME%
   com.waveset.install.UpgradePostProcess
```

Note –

- The upgrade post-process step runs in a separate Java virtual machine. The default heap size for this step is 1024 MB. If you experience out-of-memory exceptions during this step, set the maximum heap size value higher. To specify a custom value, set the `JAVA_OPTS` environment variable using the form –`Xmx`<*heap size*> where heap size is a value, such as `2048m`. An example is `-Xmx2048m`.
- The installer supports upgrading installations that have renamed, deleted, or disabled the default Configurator account.

  The installer prompts you for the user name and password to import the `update.xml` during the upgrade post process. If the user name or password is typed incorrectly, you will be prompted (up to three times) to enter the correct name or password. The error will be displayed in the text box behind it.

  For manual installation, you must provide the `-U` *username* `-P` *password* flags to pass the credentials to the `UpgradePostProcess` procedure.

---

7 **If you installed into a staging directory, create a `.war` file for deployment to your application server.**

8 **Remove the Identity Manager files from the application server work directory.**

▼ To Perform a Manual Upgrade on a UNIX Platform

Perform the following steps to upgrade Identity Manager manually on a supported UNIX platform:

1 **Stop the application server and Gateway.**

2 **Update the Identity Manager database.**

3 Set your environment.

export ISPATH=*Path-to-Install-Software* export WSHOME=*Path-to-Identity-Manager-Installation-or-Staging Directory* export TEMP=*Path-to-Temporary-Directory*

4 Run the pre-process.

```
mkdir $TEMP
cd $TEMP
jar -xvf $ISPATH/idm.war \
WEB-INF/lib/idm.jar WEB-INF/lib/idmcommon.jar
CLASSPATH=$TEMP/WEB-INF/lib/idm.jar:\
$TEMP/WEB-INF/lib/idmcommon.jar:
java -classpath $CLASSPATH -Dwaveset.home=$WSHOME \
com.waveset.install.UpgradePreProcess
```

5 Install the software.

```
cd $WSHOME
jar -xvf $ISPATH/idm.war
```

6 Run the post-process.

```
java -classpath $CLASSPATH -Dwaveset.home=$WSHOME
  com.waveset.install.UpgradePostProcess
```

Note –

- The upgrade post-process step runs in a separate Java virtual machine. The default heap size for this step is 1024 MB. If you experience out-of-memory exceptions during this step, set the maximum heap size value higher. To specify a custom value, set the `JAVA_OPTS` environment variable using the form –`Xmx`<*heap size*> where heap size is a value, such as `2048m`. An example is `-Xmx2048m`.
- The installer supports upgrading installations that have renamed, deleted, or disabled the default Configurator account.

  The installer prompts you for the user name and password to import the `update.xml` during the upgrade post process. If the user or password is typed incorrectly, you will be prompted (up to three times) to enter the correct name or password. The error will be displayed in the text box behind it.

  For manual installation, you must provide the `-U` *username* `-P` *password* flags to pass the credentials to the `UpgradePostProcess` procedure.

---

7 Change directory to `$WSHOME/bin/solaris` or `$WSHOME/bin/linux`, and set permissions on the files in the directory so that they are executable.

8 **If you installed into a staging directory, create a `.war` file for deployment to your application server.**

9 **Remove the Identity Manager files from the application server work directory.**

Troubleshooting Upgrade

If you encounter problems during the upgrade, check the upgrade log files located in the `$WSHOME/patches/logs` directory. The file names for the logs are based on a time stamp and the stage of the upgrade.

Upgrade All Gateway Instances

Upgrade every Sun Identity Manager Gateway installation in your environment. Newer versions of Identity Manager server do not work with older versions of the Gateway.

▼ To Upgrade the Identity Manager Gateway

1 **Log in to the Windows system and change to the directory where Gateway is installed.**

2 **Stop the Gateway service.**

```
gateway -k
```

3 **If using at least Windows 2000, exit all instances of the Services MMC plug-in.**

4 **Remove the Gateway service.**

```
gateway -r
```

5 **Back up and delete the existing Gateway files.**

6 **Extract the new Gateway files.**

If you are installing the newly upgraded Gateway on a system that is not the Identity Manager server, then copy the `gateway.zip` file from the Identity Manager installation package.

7 **Unpack the `gateway.zip` file into the directory where Gateway was installed.**

8 **Install the Gateway service.**

```
gateway -i
```

9 **Start the Gateway service.**

```
gateway -s
```

Upgrade All PasswordSync Instances

Unless the Release Notes specify otherwise, newly installed versions of the Identity Manager server provide limited, temporary support for older versions of PasswordSync. This support is provided so that Identity Manager can continue to run while you upgrade your PasswordSync instances. All instances of PasswordSync should be updated to the same version as Identity Manager Server as soon as possible.

To upgrade PasswordSync, you must uninstall each PasswordSync installation in your environment and reboot. Use the add/modify programs feature from the Windows Control Panel to ensure correct removal.

Replace each installation with the new PasswordSync version and reboot. Use the appropriate binary file for the operating system on which you are installing. The binary for 32-bit Windows is called `IdmPwSync_x86.msi` and the binary for 64-bit Windows is called `IdmPwSync_x64.msi`.

Note – You must reboot Windows twice: Once after uninstalling PasswordSync, and once after installing the new version. The two reboots are necessary due to the way the Windows Security Service loads the PasswordSync DLL.

For installation instructions, see "Installing and Configuring PasswordSync on Windows" in *Sun Identity Manager 8.1 Business Administrator's Guide*.

Step 8: Take Another Snapshot

After successfully upgrading the Identity Manager product, make a copy of the existing configuration objects. Also, make a copy of other object types in the repository, or copy at least a representative sampling of those objects.

The Identity Manager product upgrade does not record the changes that it makes to repository objects. If you compare this snapshot to the snapshot that you took before the upgrade, you can easily detect any changes made to repository objects during the upgrade.

Step 9: Analyze the Changes

You must analyze the changes made by the Identity Manager product upgrade, and update your configurations and customizations accordingly. For example:

- If you modified any JSP files or stylesheets, you must merge these changes into the new JSP files or stylesheets.

- If your Identity Manager application baseline includes Identity Manager product JARs and third-party JARs, you might have to update these JARs in the baseline. Your baseline should also include the SQL scripts that are used to create or update your database tables.
- If you modified any of the default Identity Manager objects (such as the Default User Form), the upgrade process moves those objects into the `savedObjects` directory. To facilitate future upgrades, rename the modified objects with a custom name, and reference that name in the `SystemConfiguration` object.
- If you extracted `WPMessages.properties` to the `/config` directory and customized any of the messages, you must extract and reapply these customizations.

You must carefully analyze changes made to repository objects during the Identity Manager product upgrade. For example:

- If the Identity Manager product upgrade modified configuration objects that are in your source-control baseline, you must merge these changes into your configuration baseline. For more information, see "Step 14: Merge Changes Back Into Source Control" on page 49.
- If the Identity Manager product upgrade modified configuration objects that are not currently in your baseline, you must add these objects to your application baseline. If you do not add these configuration objects to your application baseline, then you must make other plans to incorporate these changes, such as including the appropriate objects or commands within the subset of `update.xml` that your upgrade procedure imports in each environment.

Tip – You might decide that you can safely ignore these object changes, but in most cases it is considered a best practice to add these configuration objects to your baseline.

- If the Identity Manager product upgrade modified objects in the repository that are not configuration objects, then these objects should not become part of your source-control baseline. For example, the Identity Manager `update.xml` file might refresh `TaskInstance` objects, `User` objects, `Account` objects, or `Entitlement` objects.

  Identity Manager Engineering generally avoids updating these object types because there can be so many instances of each type, but in some cases changes are necessary or justified. In such cases, include executing an appropriate subset of the Identity Manager `update.xml` file in your baseline and in your upgrade process. Use this `update.xml` subset to update repository objects that are not part of your baseline.

After upgrading, restore any customized files and objects.

Restoring Customized Files

During the upgrade, Identity Manager automatically copies all customized files, such as JSP and HTML files, into the following directory:

```
$WSHOME/patches/Sun_Java_System_Identity_Manager_Version_Date_/savedFiles
```

The following table describes the files in this directory.

TABLE 3–1 `savedFiles` Directory File Structure

File Name	Description
`changedFileList`	File containing a list of all saved customized files. This file also contains a list of files (installed with your older version of Identity Manager) that will be overwritten when files of the same name are installed during upgrade.
`notRestoredFileList`	File containing a list of all customized files that are *not* restored during the upgrade process.
`notInstalledFileList`	File containing a list of newer version files that are *not* installed during the upgrade process.

The upgrade might add some files that were also installed with your original Identity Manager installation. Before overwriting the older files, Identity Manager automatically saves them in the `savedFiles` directory. See the `changedFileList` file for a list of these files.

Identity Manager automatically restores most of the files listed in `changedFileList` during the upgrade process, but does not restore all of them. See the `notRestoredFileList` for a list of these files. When restoring customized files, Identity Manager overwrites the newer version of the files that were installed during the upgrade.

You might have to manually restore some of your file customizations. Review the `notRestoredFileList` file to see a list of the files that were *not* restored during upgrade. If you must manually restore any customized files, edit the new file that was installed during the upgrade to incorporate your customizations, and then save the newly edited file.

Restoring Customized Objects

If you have configured your form and process mappings in the system configuration, you will not have to restore those object customizations after the upgrade. If you have customized objects that are not listed in the system configuration, then you must manually restore these objects by importing the XML for these objects.

As a safety measure, Identity Manager automatically saves many of the commonly customized objects to files when you import `update.xml`. These files are saved to subdirectories in the `WEB-INF/savedObjects` directory. These subdirectories are named with a time stamp of the time at which the import was performed.

Importing `update.xml` can create up to three subdirectories in the `savedObjects` directory. You can manually import the object XML files to restore object customizations.

Step 10: Rebuild All Custom Java Classes

You must rebuild all of your custom Java classes against the new product libraries. For example, you must rebuild any new JAR files or application server libraries.

If recompiling produces deprecation warnings, analyze each deprecation message, and read the *Sun Identity Manager 8.1 Release Notes* to determine whether you can resolve the deprecation issue immediately. If you cannot resolve the deprecation issue immediately, add an item to your project plan to resolve the issue in the future.

Note – Identity Manager does not support deprecated APIs indefinitely. Deprecated classes and methods are generally removed in the next major product release.

Step 11: Make Necessary Changes in XPRESS

Make any forms, rules, and workflows changes in XPRESS.

The forms, rules, and workflows supplied in new Identity Manager product versions are generally backward-compatible with older forms, rules, and workflows. The most common type of change required is to change invocations of Identity Manager Workflow Services or Form Utility methods.

Note – For information about release-specific changes to Workflow Services or Form Utility methods, see the Identity Manager Release Notes for the release to which you are upgrading.

Step 12: Test Your Identity Manager Application in the Development Environment

Restart the application server and test your Identity Manager application at least minimally to confirm that at least the basic functions are working as expected.

You must redeploy your web applications after upgrading Identity Manager because most application servers cache the `web.xml` file.

▼ To Redeploy Your Web Application After Upgrading

If you are using the Sun GlassFish™ Enterprise Server, for example, you would perform the following steps to redeploy a web application after upgrading Identity Manager:

1. **Log in to the GlassFish administrator interface.**

2. **Choose Applications > Web Applications from the menu bar.**

3 **Locate your web application and click the Redeploy link.**

4 **Click the button next to the Local Packaged File or Directory That Is Accessible From the Application Server option.**

5 **Click the Browse Folders button and select the top-level folder for your installation.**

For example:

```
C:\Sun\AppServer\domains\domain1\applications\j2ee-modules\idm
```

6 **Click OK.**

7 **Restart the application server.**

Step 13: Restart Active Sync and Reconciliation

After successfully upgrading, you must restore the original settings for any Active Sync processes and reenable any scheduled reconciliations (if applicable).

Tip – Step 13 is optional, but performing this step is considered a best practice when upgrading the Production environment.

Also, if you perform Step 13 in your Production environment, make it a standard step when upgrading all of your other environments.

Step 14: Merge Changes Back Into Source Control

Note – Merging changes back into source control is specifically listed here as a separate step to highlight its importance. In actual practice, you can merge changes back into source control as you perform Steps 9 through 12.

When merging changes back into source control, you must:

- Verify that all existing customizations are tagged and stored in the version control system.
- Check in all new customizations after completing the test upgrade cycle, including the following modified items:
  - Database table scripts
  - Repository objects
  - File system objects, such as JSP files and images

Task 7: Reset Your Test Environment

To perform controlled testing, you must reset your Test environment so that it corresponds to your Production environment as closely as possible.

Step 1: Reset Your Platform to Match Production

To reset your Test environment, ensure the following:

- The platform in the Test environment matches your current Production environment. The platform includes the application server, Repository DBMS, and JDK version. See "Step 1: Document Your Platform" on page 22.
- The Identity Manager application image in your Test environment corresponds to the application baseline for your current Production environment.
- The database table definitions in the Test environment match those in the Production environment.
- Resources and other integrated applications match those in the Production environment.

  If real test resources do not exist, you can create simulated resources for the functional test.

Every time you promote an image of your Identity Manager application from the Development environment, you must test your cumulative upgrade procedure. If the upgrade procedure appears to be successful, execute your test plan.

Step 2: Set Up for Functional Testing

To prepare for functional testing, you must create a Test environment that supports controlled testing of your Identity Manager application.

You might want to simulate some aspects of the Production environment, but the primary goal is to verify that the application works as expected. Achieving this goal might require that you load controlled datasets rather than perfectly realistic ones.

Load test data into your database tables that supports execution of the test cases in your test plan. Ideally, the database tables would also contain data similar to the data in your Production environment.

Task 8: Execute Your Upgrade Procedure

Upgrading a Test environment requires only a subset of the steps that you performed when upgrading your Development environment. For example, you do not have to detect changes or update source control. The updated baseline for your Identity Manager application already contains those changes.

Before upgrading any targeted environments, you must generate an image of your Identity Manager application that is appropriate for that environment. The baseline, and therefore the image, contains the following:

- SQL scripts that update the database tables, file system objects, and repository objects
- An appropriate subset of `update.xml` to update repository objects that are not in your baseline

Step 1: Stop Active Sync and Reconciliation

Set any Active Sync processes to start manually and, if applicable, disable any scheduled reconciliations until the upgrade is complete and appears to be successful.

Tip – Step 1 is optional, but performing this step is considered a best practice when upgrading the Production environment.

Also, if you perform Step 1 in your Production environment, make it a standard step when upgrading in all of your other environments.

Step 2: Stop the Identity Manager Application

Quiesce your Identity Manager application and make it unavailable to all administrators and end users.

Step 3: Back Up Your Identity Manager Application

Make a copy of your existing database and Identity Manager file structure.

Backing up the database and file structure enables you to reinstate your working environment, if necessary.

Note – Always back up the Identity Manager database and file system before applying any Identity Manager patches, service packs, or hotfixes and before going through any major upgrades.

You can use third-party backup software or a backup utility supplied with your system to back up the Identity Manager file system. To back up your database, see your database documentation for recommended backup procedures.

▼ To Back Up Your Identity Manager Application

1. **Shutdown or idle Identity Manager.**
2. **Use your backup utilities to back up your database and the file system where you installed Identity Manager.**

Step 4: Remove Hotfixes

Remove any hotfix class files from your `WEB-INF/classes` directory.

Hotfix class files generally work only with the specific version of the Identity Manager product for which the hotfix was delivered.

Step 5: Change `TaskDefinition` Objects

You might find it necessary to upgrade a Production environment that contains executing task instances. Unfortunately, upgrading an Identity Manager `TaskDefinition` object in the repository can corrupt executing task instances that depend on the `TaskDefinition` object. This possibility is a particularly important consideration in a Production environment where people are depending on those tasks to complete correctly and to perform their business functions.

Although it is easiest to have users complete their tasks or terminate still-executing tasks prior to upgrade, these options are not always feasible.

If your Production environment might contain executing task instances when you upgrade, be sure that your upgrade procedure describes how to address these instances.

Tip – Rename `TaskDefinition` objects when upgrading in each environment. Use the following process to upgrade `TaskDefinition` objects in your Production environment:

1. From the Identity Manager console, rename the current `TaskDefinition` to include a time stamp.
2. Load the new `TaskDefinition`.

Caution – Problems might occur if you change activities or actions.

Note that you cannot modify any `TaskDefinitions` that correspond to live `TaskInstances`. Identity Manager does not allow you to make these modifications.

Step 6: Update Your Platform

If the target Identity Manager product version requires platform changes, you must make these changes before upgrading the Identity Manager product.

Step 7: Upgrade Your Identity Manager Application

To upgrade your Identity Manager application, you might be required to do the following:

- "Update Your Database Table Definitions" on page 54
- "Promote the Identity Manager Application" on page 54
- "Import a Subset of `update.xml`" on page 54
- "Upgrade All Gateway Instances" on page 55
- "Upgrade All PasswordSync Instances" on page 55

Note – About Data Sources

If you use a JDBC data source defined in your application server as your Identity Manager repository location, be aware that this data source might not work outside the application server. In other words, a JDBC data source provided by an application server might be available for use only by web applications that run in that container.

The Identity Manager product upgrade process runs outside the application server, just like the Identity Manager console. Therefore, in each environment where Identity Manager normally uses a data source, your upgrade procedure might need to include steps to switch to a JDBC DriverManager connection.

You can temporarily replace the `ServerRepository.xml` file that specifies a data source with another `ServerRepository.xml` file that specifies a JDBC DriverManager connection. Restore the original `ServerRepository.xml` file as a subsequent step in your upgrade procedure.

Alternatively, you can expand the Identity Manager application `WAR` file onto the file system, specify `WSHOME` as the file system location, and use this "side" environment to perform a manual upgrade process or to perform any step that requires a console, such as importing a subset of `update.xml` or renaming `TaskDefinition` objects.

If additional setup is required for your custom integrations in each environment, perform the additional setup as part of this step.

Update Your Database Table Definitions

Verify that your Identity Manager application image includes any SQL scripts needed to update your database table definitions, and that these SQL scripts have been modified to fit your environment.

If your image does not include these SQL scripts, ensure that your upgrade procedure specifically describes the modifications required for each environment.

Promote the Identity Manager Application

Promote the Identity Manager application image into your Test environment. Your application image must include the target Identity Manager product version, your updated configuration, and your customizations.

Import a Subset of `update.xml`

You must import the `update.xml` file to update the repository objects that are not managed as part of your Identity Manager application baseline.

Tip – Use only one Identity Manager server to import `update.xml` and have only one Identity Manager server running during the upgrade.

If you start any other Identity Manager servers during the upgrade process, you must stop and restart those servers before making them available again.

Upgrade All Gateway Instances

Upgrade every Sun Identity Manager Gateway installation in your environment. See "To Upgrade the Identity Manager Gateway" on page 44.

Caution – Newer versions of Identity Manager server will not work with older versions of the Gateway. All Gateway and Identity Manager Server installations should be updated within the same maintenance window.

Upgrade All PasswordSync Instances

Upgrade every PasswordSync installation in your environment. See"Upgrade All PasswordSync Instances" on page 45.

Unless the Release Notes specify otherwise, newly installed versions of the Identity Manager server provide limited, temporary support for older versions of PasswordSync. This support is provided so that Identity Manager can continue to run while you upgrade your PasswordSync instances. All instances of PasswordSync should be updated to the same version as Identity Manager Server as soon as possible.

Step 8: Test Your Identity Manager Application

You must redeploy your web applications after upgrading Identity Manager because most application servers cache the `web.xml` file.

Restart the application server and test your Identity Manager application at least minimally to verify that the basic functions are working as expected.

▼ To Redeploy Your Identity Manager Application

If you are using the Sun GlassFish Enterprise Server, perform the following steps to redeploy Identity Manager.

1. **Log in to the GlassFish administrator interface.**

2. **Choose Applications > Web Applications from the menu bar.**

3 **Locate your web application and click the Redeploy link.**

4 **Click the button next to the Local Packaged File or Directory That Is Accessible From the Application Server option.**

5 **Click the Browse Folders button and select the top-level folder for your installation.**

For example:

```
C:\Sun\AppServer\domains\domain1\applications\j2ee-modules\idm
```

6 **Click OK.**

7 **Restart the application server.**

Step 9: Restart Active Sync and Reconciliation

After successfully completing the upgrade, restore the original settings for any Active Sync processes and for any scheduled reconciliations.

Tip – Step 9 is optional, but performing this step is considered a best practice when upgrading the Production environment.

Also, if you perform Step 9 in your Production environment, make it a standard step when upgrading all of your other environments.

Step 10: Restart the Identity Manager Application

Restart the Identity Manager application to make the application available again to administrators and end users.

Task 9: Perform Functional Testing

Testing in the Test environment is crucial before you deploy the Development upgrade image into your Production environment.

▼ To Test Your Test Environment After Upgrading

1 **Execute your complete test plan, including any automated tests.**

2 **Fix any problems and incorporate the fixes into the source-control baseline in your Development environment.**

3 **Repeat the process of resetting your Test environment, upgrading your Test environment, and retesting your Identity Manager application.**

◆ ◆ ◆ CHAPTER 4

User Acceptance Testing

The User Acceptance Testing phase consists of the following tasks, which are very similar to the tasks described in Chapter 3, “Developing and Testing Your Upgrade”:

- “Task 10: Reset Your QA Environment” on page 60
- “Task 11: Upgrade Your QA Environment” on page 60
- “Task 12: Perform User Acceptance Testing” on page 61

This document describes user acceptance testing as a separate upgrade phase for these reasons:

- The QA environment must replicate the Production environment as closely as possible, which allows you to test the upgrade procedure against realistic data values and against a realistic volume of data. For more information, see “Task 10: Reset Your QA Environment” on page 60 and “Task 11: Upgrade Your QA Environment” on page 60.
- A skip-level upgrade greatly affects how you develop and test your upgrade, but does not affect user acceptance testing. For more information, see Chapter 6, “Skip-Level Upgrade Considerations.”
- User acceptance testing verifies that your Identity Manager application meets the needs of the people who use or who are intended to use the application. For more information, see “Task 12: Perform User Acceptance Testing” on page 61.

Task 10: Reset Your QA Environment

Reset your QA environment to mimic your Production environment as closely as possible, being sure to use the same hardware and software versions used in the Production environment.

Specifically, you want to replicate the following in your Production environment:

- Platform, including the following:
  - Application server
  - Database server
  - Gateway server
  - Web Server (optional component)
  - Common client machine with the corporate image and same browser
- Identity Manager product version
- Identity Manager application configurations and customizations
- Database tables and actual data, including data values and the volume of data found in the Production environment
- Resources and other integrated applications

  Configure resources and other integrated applications in the QA environment to have the same general configuration as those used in the Production environment. For example, replicate Active Directory domains, organizations, groups, numbers, and the format of users from Production to the QA environment.

  Also, be sure to configure integrated applications the same way in both the QA environment and the production environment, and make sure that these applications contain a copy of recent production data or very similar data.

Task 11: Upgrade Your QA Environment

The process for upgrading your QA environment is very similar to the process described in "Task 6: Upgrade Your Development Environment" on page 34.

Other than a few different parameter values, such as host names and connection information, your procedure for updating your QA environment must contain exactly the same steps as your procedure for upgrading the Test environment.

Because your QA environment is supposed to replicate the Production environment as closely as possible, you must monitor the upgrade closely for any errors caused by data values, data volumes, or platform considerations that were not fully tested in the Test environment.

To help you plan the final phase of your upgrade, upgrading your Production environment, carefully measure how long it takes you to execute various steps in your upgrade procedure.

Task 12: Perform User Acceptance Testing

Performing user acceptance testing is comparable to performing "Task 9: Perform Functional Testing" on page 56. In fact, you might choose to execute your test plan or a subset of the plan in this simulated Production environment.

The main difference between these two tasks is that for user acceptance testing, you must arrange for the people who actually use your Identity Manager application to test it with realistic data.

Though it might be difficult to schedule user participation with people distributed across your organization, user acceptance testing is usually quite valuable. Giving users an advanced look at the next version of your Identity Manager application generally helps maintain productivity and encourages adoption. Conducting user acceptance testing also demonstrates that the deploying organization is proactive and responsive to the needs of the people who are using the application.

User acceptance testing often uncovers problems and clarifies requirements for your Identity Manager application. The users might also find issues that are specific to aspects of your platform that were not tested in other environments. Although most developers prefer to find problems in earlier phases, finding problems during user acceptance testing is still much better than finding problems after you upgrade the Production environment.

Even if you ultimately decide to go into production with issues or limitations, you will know about these issues or limitations ahead of time. Key users are then prepared to communicate the problems and any workarounds to other users of the application.

▼ To Address Issues Discovered In User Acceptance Testing

As with Task 9, if you have to fix any problems, you must do the following:

1. **Incorporate the fixes back into your Development environment's source-control baseline.**
2. **Reset your Test environment.**
3. **Upgrade your Test environment.**
4. **Retest your Identity Manager application.**
5. **Generally, repeat User Acceptance Testing.**

   You might decide that incremental testing is sufficient.

◆ ◆ ◆ CHAPTER 5

Upgrading Your Production Environment

Although you perform only one task to upgrade your Production environment, and the process for upgrading Production is very similar to the processes described in "Task 6: Upgrade Your Development Environment" on page 34 and "Task 11: Upgrade Your QA Environment" on page 60, upgrading Production is specifically described separately to highlight its importance.

This chapter covers the following topics:

- "Special Considerations for Production" on page 63
- "Task 13: Upgrade Your Production Environment" on page 64

Special Considerations for Production

Your upgrade procedure probably includes steps that are important only when you are upgrading your Production environment. For example, your upgrade procedure might include:

- Scheduling an outage for your Identity Manager application
- Scheduling database administrator support
- Notifying users before taking the application offline
- Shutting down specific resources, processes, or applications that are used only in your Production environment

Upgrading your Production environment might also pose special considerations that require *advanced techniques*. For example, to minimize the downtime of your Identity Manager application, you might want to upgrade a separate server and a separate database in a parallel Production instance. After upgrading the parallel instance, you might want to make the original Production instance unavailable, synchronize from the actual Production system to the parallel instance any data that changed during the time that the parallel instance was being upgraded, and then switch over to the parallel Production instance.

Note – Advanced techniques such as these are beyond the scope of this document. If you have any special considerations or technical questions related to upgrading your Production environment, contact your Sun Professional Services to request assistance.

Task 13: Upgrade Your Production Environment

The procedure for updating the Production environment must contain the same steps as your procedure for updating the QA environment. Some specific parameter values, such as host names and connection information, can be different.

If there is any significant difference between the procedure for upgrading the QA environment and the procedure for upgrading the Production environment, then document the specific reasons for this variance. For example, it might not be feasible to simulate certain aspects of the Production environment in the QA environment. By documenting this variance, you allow the people involved in your Upgrade Project to make an informed judgment as to the risk posed by this variance.

CHAPTER 6

Skip-Level Upgrade Considerations

This chapter describes the special considerations related to performing a skip-level upgrade, and covers the following topics.

Skip-Level Upgrade Overview

You develop a *skip-level* (or *multi-hop*) upgrade to update your Production environment to a version of the Identity Manager product that is beyond the next major release from your current version. For example, if you are currently using Version 6.0 of Identity Manager and want to upgrade directly to Version 8.1, this upgrade path would require a skip-level upgrade.

Upgrading several versions normally requires a series of upgrades. However, many customers want to minimize the number of upgrades in the Production environment, which in turn minimizes the *downtime* of your Identity Manager application, minimizes the cost of retesting the Identity Manager application, and minimizes the cost of retraining the people using the Identity Manager application.

Developing a single upgrade procedure to update a target Identity Manager version is technically possible and some customers find it feasible. The key point to understand is that, even though you are performing a skip-level upgrade in your Production environment, you still must perform each *hop* in your Development environment. For example, you must upgrade Identity Manager from Version 7.0 to Version 7.1 to Version 7.1.1 to Version 8.0 to Version 8.1 in your Development environment. After each hop, you build up a set of artifacts that is cumulative.

The most common approach for a skip-level upgrade is to update your application baseline with configurations and customizations that have been updated to work with each Identity

Manager version on the path to your target version of Identity Manager. Your baseline also includes a cumulative script to update databases and a cumulative subset of `update.xml`. The net effect is that you develop a single upgrade procedure that makes changes that are equivalent to the changes that performing the series of upgrades would have done.

Performing a skip-level upgrade is more complex than a standard or *single-hop* upgrade. Skip-level upgrades require more technical insight into the mechanisms used by the Identity Manager product upgrade. A skip-level upgrade also requires closer analysis of the upgrade content for each Identity Manager product version in the upgrade path. Closer analysis allows you to produce artifacts that are properly cumulative and yet minimal. For example, if you simply combine all of the database table upgrade scripts into one script or if you combine the subsets of `update.xml` from each step into one subset, then your upgrade might perform a great deal of redundant processing.

This chapter describes special considerations for a skip-level upgrade. If you are uncertain how to proceed or if these considerations seem unclear to you, contact Sun Professional Services to request assistance with planning your upgrade.

Phases of a Skip-Level Upgrade

At a high level, the steps you perform for a skip-level upgrade are the same as those performed for a regular upgrade. However, some of these steps are enhanced and some steps are repeated for a skip-level upgrade.

The following figure illustrates the skip-level upgrade process.

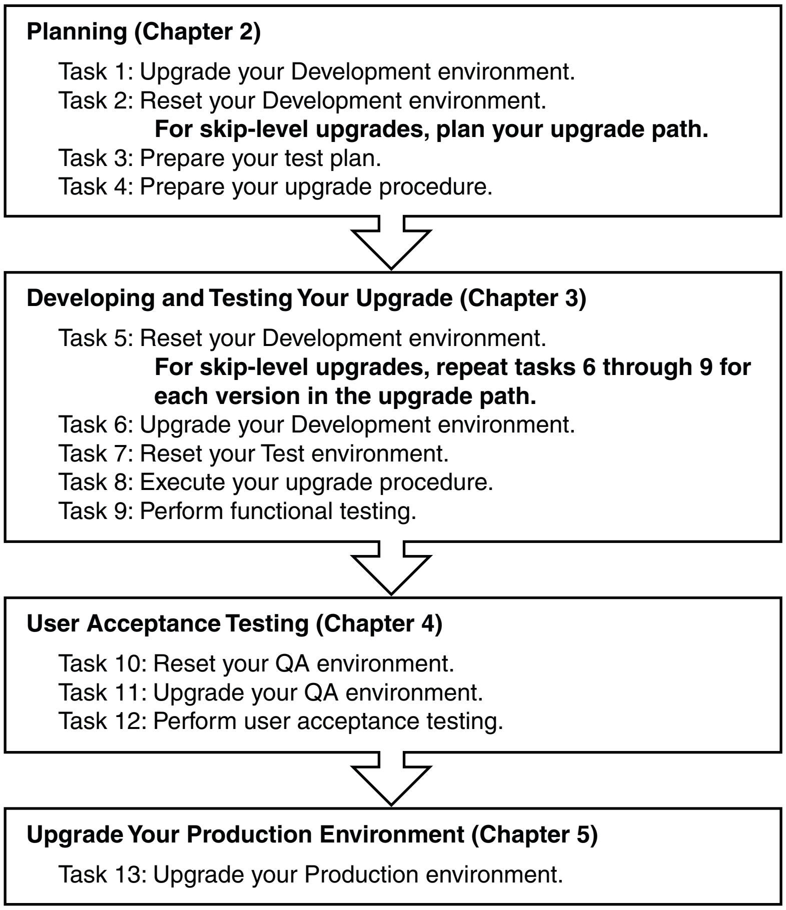

Planning (Chapter 2)
Task 1: Upgrade your Development environment.
Task 2: Reset your Development environment.
For skip-level upgrades, plan your upgrade path.
Task 3: Prepare your test plan.
Task 4: Prepare your upgrade procedure.
Developing and Testing Your Upgrade (Chapter 3)
Task 5: Reset your Development environment.
For skip-level upgrades, repeat tasks 6 through 9 for each version in the upgrade path.
Task 6: Upgrade your Development environment.
Task 7: Reset your Test environment.
Task 8: Execute your upgrade procedure.
Task 9: Perform functional testing.
User Acceptance Testing (Chapter 4)
Task 10: Reset your QA environment.
Task 11: Upgrade your QA environment.
Task 12: Perform user acceptance testing.
Upgrade Your Production Environment (Chapter 5)
Task 13: Upgrade your Production environment.

FIGURE 6–1 Skip-Level Upgrade Process

For planning purposes, the first difference between a regular upgrade and a skip-level upgrade is in how you perform "Task 2: Choose the Target Identity Manager Version" on page 28. When choosing the target Identity Manager version, you must plan your upgrade path.

The next, and biggest, difference between the two upgrade processes is that you must perform "Task 6: Upgrade Your Development Environment" on page 34 through "Task 9: Perform Functional Testing" on page 56 *for each stop in the upgrade path*. You must upgrade your Development environment and your application baseline for the Identity Manager product version at each stop in the upgrade path. You probably also want to retest after each stop in the upgrade path, which requires resetting your Test environment and promoting your Identity Manager application to the Test environment after each stop in the upgrade path.

Tip – You might think that retesting after each stop on the upgrade path is unnecessary, but testing the upgrade process and testing the Identity Manager application at each stop on the upgrade path minimizes your risk because you can identify problems quickly. Working with a smaller set of changes makes it much easier to isolate the cause of any problems.

Task 2: Choose the Target Identity Manager Version (Enhanced)

When performing a skip-level upgrade, you must plan your upgrade path after deciding which Identity Manager version is your target.

Note – For information about Identity Manager upgrade paths, see "Upgrade Paths and Support Policies" in *Sun Identity Manager 8.1 Release Notes*.

The standard upgrade processes supplied with each full Identity Manager release generally upgrade an existing installation from any version of the previous major release.

Remember that each stop in the upgrade path requires a different version of the Identity Manager product. To plan your skip-level upgrade, you must read the Release Notes provided for the Identity Manager product version at each stop in the upgrade path.

For example, if you are upgrading Identity Manager from version 6.0 to version 8.1, you must read the Release Notes for the following versions of Identity Manager:

- Version 6.0 and all its service packs
- Version 7.0 and all its service packs
- Version 7.1 and all its patches
- Version 8.0 and all its patches
- Version 8.1

Task 6: Upgrade Your Development Environment (Repeated)

When performing a skip-level upgrade, you must repeat Task 6 once for each stop in the upgrade path.

Perform Steps 1–14 as described in "Task 6: Upgrade Your Development Environment" on page 34 *for each Identity Manager product version* to which you must upgrade to reach your Identity Manager target version.

Note – Of these steps, only "Step 9: Analyze the Changes" on page 45 must be significantly enhanced for a skip-level upgrade. See "Step 9: Analyze the Changes for a Skip-Level Upgrade" on page 69 for the enhanced instructions.

Step 9: Analyze the Changes for a Skip-Level Upgrade

You must analyze the changes made by the Identity Manager product upgrade.

As you iteratively upgrade your Development environment, you iteratively update the baseline for your Identity Manager application, including the following:

- Configuration objects
- JSP files
- Identity Manager product JARs
- Third-party JARs

Your baseline must also include SQL scripts to create or update database tables and a subset of `update.xml` to update repository objects that are not included in your baseline.

Each iteration that includes a sample database table upgrade script requires changes to your overall upgrade procedure. You can run the database table upgrade scripts in the correct order or concatenate the scripts, but you must modify each sample script appropriately for your environment.

You might find that it is more convenient, more efficient, and ultimately safer to write a single database table upgrade script that is cumulative. In other words, write a single script that combines all of the processing that would have been done by each of the individual database table upgrade scripts if you executed those scripts in the proper order.

Executing a single database upgrade script simplifies the upgrade procedure and gives you the opportunity of eliminating redundant processing, such as creating indexes for one version of Identity Manager and then later dropping and re-creating the same indexes for another version of Identity Manager.

You can also identify an appropriate subset of the Identity Manager `update.xml` in each iteration that is required to update objects in the Identity Manager repository that are not managed as part of the baseline. For a skip-level upgrade, you must ensure that this subset of the Identity Manager `update.xml` is cumulative.

Note – If you are developing a skip-level upgrade, you must be sure to add any configuration objects that were changed by an *updater* to your Identity Manager baseline.

An updater is a program supplied with Identity Manager that updates configuration objects. The updater is invoked by an `ImportCommand` within `update.xml` or within a file that `update.xml` includes. An updater generally works only with the version of Identity Manager with which it was shipped. Because you are writing a "skip-level" upgrade, the updater probably will not work with the target version of Identity Manager. Adding any changed configuration object to your baseline is by far the safest approach.

CHAPTER 7

Assessment Worksheets

Having a good understanding of your current Identity Manager installation and the latest Identity Manager version helps you determine an appropriate upgrade method and strategy.

Use the worksheets provided in this chapter to record important configuration data, and then use this data to prepare for and choose an upgrade path.

- "Platform Inventory" on page 72
- "Identity Manager Installation" on page 74
- "Custom Components" on page 75

Note – If you are using the Identity Manager IDE or an older form of CBE, their automated tools already capture much of this information for you.

Platform Inventory

Record platform inventory information in Table 7–1.

TABLE 7–1 Platform Inventory Information

Platform Component	Your Information
Application server	Version and Service Packs _______________ _______________ _______________ Server Platform Version and Service Packs _______________ JDK _______________
Database server	Version and Service Packs _______________ _______________ _______________ Server Platform Version and Service Packs _______________
Sun Identity Manager Gateway server	Version _______________
Java Runtime Environment (JRE)	Version and Vendor _______________
Web servers	Version and Service Packs _______________ _______________ _______________

Record resource information in Table 7–2.

TABLE 7–2 Resource Information

Resource	Name	Version and Service Packs
Resource A	______________________	______________________ ______________________ ______________________
Resource B	______________________	______________________ ______________________ ______________________
Resource C	______________________	______________________ ______________________ ______________________
Resource D	______________________	______________________ ______________________ ______________________
Resource E	______________________	______________________ ______________________ ______________________
Resource F	______________________	______________________ ______________________ ______________________
Resource G	______________________	______________________ ______________________ ______________________
Resource H	______________________	______________________ ______________________ ______________________
Resource I	______________________	______________________ ______________________ ______________________

Identity Manager Installation

Record Identity Manager installation information in Table 7–3.

TABLE 7–3 Identity Manager Installation Information

Identity Manager	Your Information
Installed version	______________________________
Installed service packs	______________________________ ______________________________ ______________________________ ______________________________ ______________________________
Installed hotfixes	______________________________ ______________________________ ______________________________ ______________________________ ______________________________ ______________________________ ______________________________ ______________________________

Custom Components

Record custom work information in Table 7–4.

Custom work consists of any component that has been modified for your installation.

TABLE 7–4 Custom Work Information

Component	Your Information
File-System Objects	
Modified JSP files	____________________ ____________________ ____________________ ____________________
Modified `Waveset.properties` file	____________________ ____________________ ____________________ ____________________
Modified `wpmessages.properties` file	____________________ ____________________ ____________________ ____________________
Other customized property files	____________________ ____________________ ____________________ ____________________
Customized resource adapters (and other custom Java)	____________________ ____________________ ____________________ ____________________ ____________________ ____________________ ____________________

TABLE 7–4 Custom Work Information *(Continued)*

Component	Your Information
Modified stylesheets	______ ______ ______ ______
Repository Objects	
Modified forms	______ ______ ______ ______
Modified workflows	______ ______
Modified email templates	______ ______ ______ ______
Custom repository schema	______ ______
Other Custom Repository Objects	
Admin groups	______ ______
Admin roles	______ ______
Configurations	______ ______
Policies	______ ______
Provisioning tasks	______ ______
Remedy configurations	______ ______

TABLE 7–4 Custom Work Information *(Continued)*

Component	Your Information
Resource actions	______________________________ ______________________________
Resource forms	______________________________ ______________________________
Roles	______________________________ ______________________________
Rules	______________________________ ______________________________
Task definitions	______________________________ ______________________________
Task templates	______________________________ ______________________________
User forms	______________________________ ______________________________ ______________________________ ______________________________

Index