Sun Java™ System

Identity Server Migration Guide

2004Q2

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

Part No: 817-5708-10

Contents

List of Figures

About This Guide

The *Sun Java™ System Identity Server Migration Guide* describes how to upgrade a deployment to Sun Java System Identity Server 2004Q2 (formerly Sun™ ONE Identity Server) and to migrate user data from an existing Sun Java System Directory Server. This preface contains the following sections:

Audience for This Guide

This *Identity Server Migration Guide* is intended for system administrators and software developers who are implementing an integrated identity management and web access platform using the Sun Java Enterprise System. Readers should be familar with the following technologies:

- Lightweight Directory Access Protocol (LDAP)
- Java™ technology
- JavaServer Pages™ (JSP) technology
- HyperText Transfer Protocol (HTTP)
- HyperText Markup Language (HTML)
- eXtensible Markup Language (XML)

Because Sun Java System Directory Server is used as the data store in an Identity Server deployment, administrators should also be familiar with the Directory Server documentation, which is available on the following Web site:

http://docs.sun.com/coll/DirectoryServer_04q2

Identity Server Documentation Set

The Identity Server documentation set is separated into two core sets of manuals: the Sun Java System Identity Server 2004Q2 core application manuals and the Sun Java System Identity Server Policy Agents books.

Identity Server Core Documentation

The Identity Server documentation set contains the following titles:

The Identity Server documentation set contains the following titles:

- *Technical Overview* (http://docs.sun.com/doc/817-5706) provides a high-level overview of how Identity Server components work together to consolidate identity management and to protect enterprise assets and web-based applications. It also explains basic Identity Server concepts and terminology.
- *Migration Guide* (this document) provides details on how to migrate existing data and Sun Java System product deployments to the latest version of Identity Server. (For instructions about installing Identity Server and other products, see the *Sun Java Enterprise System 2004Q2 Installation Guide* (http://docs.sun.com/doc/817-5760).
- *Administration Guide* (http://docs.sun.com/doc/817-5709) describes how to use the Identity Server console as well as manage user and service data via the command line.
- *Deployment Planning Guide* (http://docs.sun.com/doc/817-5707) provides information on planning an Identity Server deployment within an existing information technology infrastructure.
- *Developer's Guide* (http://docs.sun.com/doc/817-5710) offers information on how to customize Identity Server and integrate its functionality into an organization's current technical infrastructure. It also contains details about the programmatic aspects of the product and its API.

- *Developer's Reference* (http://docs.sun.com/doc/817-5711) provides summaries of data types, structures, and functions that make up the public Identity Server C APIs.
- *Federation Management Guide* (http://docs.sun.com/doc/817-6362) provides information about Federation Management, which is based on the Liberty Alliance Project.
- The *Release Notes* (http://docs.sun.com/doc/817-5712) will be available online after the product is released. They gather an assortment of last-minute information, including a description of what is new in this current release, known problems and limitations, installation notes, and how to report issues with the software or the documentation.

Updates to the *Release Notes* and links to modifications of the core documentation can be found on the Identity Server page at the Sun Java System 2004Q2 documentation web site (http://docs.sun.com/prod/entsys.04q2). Updated documents will be marked with a revision date.

Identity Server Policy Agent Documentation Set

Policy agents for Identity Server documents are available on this Web site:

http://docs.sun.com/coll/S1_IdServPolicyAgent_21

Policy agents for Identity Server are available on a different schedule than the server product itself. Therefore, the documentation set for the policy agents is available outside the core set of Identity Server documentation. The following titles are included in the set:

- *Policy Agents For Web and Proxy Servers Guide* documents how to install and configure an Identity Server policy agent on various web and proxy servers. It also includes troubleshooting and information specific to each agent.
- *J2EE Policy Agents Guide* documents how to install and configure an Identity Server policy agent that can protect a variety of hosted J2EE applications. It also includes troubleshooting and information specific to each agent.
- The *Release Notes* will be available online after the set of agents is released. There is generally one *Release Notes* file for each agent type release. The *Release Notes* gather an assortment of last-minute information, including a description of what is new in this current release, known problems and limitations, installation notes, and how to report issues with the software or the documentation.

Updates to the *Release Notes* and modifications to the policy agent documentation can be found on the Policy Agents page at the Sun Java System documentation web site. Updated documents will be marked with a revision date.

Your Feedback on the Documentation

Sun Microsystems and the Identity Server technical writers are interested in improving this documentation and welcomes your comments and suggestions. Use the following web-based form to provide feedback to us:

http://www.sun.com/hwdocs/feedback/

Please provide the full document title and part number in the appropriate fields. The part number can be found on the title page of the book or at the top of the document, and is usually a seven or nine digit number. For example, the part number of the *Sun Java System Identity Server 2004Q2 Migration Guide* is 817-5708-10.

Documentation Conventions Used in This Guide

In the Identity Server documentation, certain typographic conventions and terminology are used. These conventions are described in the following sections.

Typographic Conventions

This book uses the following typographic conventions:

- *Italic type* is used within text for book titles, new terminology, emphasis, and words used in the literal sense.
- `Monospace font` is used for sample code and code listings, API and language elements (such as function names and class names), filenames, pathnames, directory names, HTML tags, and any text that must be typed on the screen.
- *Italic serif font* is used within code and code fragments to indicate variable placeholders. For example, the following command uses *filename* as a variable placeholder for an argument to the `gunzip` command:

  `gunzip -d` *`filename`*`.tar.gz`

Terminology

Below is a list of general terms used in the Identity Server documentation set:

The following terms are used in the Identity Server documentation set:

- *Identity Server* refers to Identity Server and any installed instances of the Identity Server software.
- *Policy and Management services* refers to the collective set of Identity Server components and software that are installed and running on a dedicated deployment container such as a web server.
- *Directory Server* refers to an installed instance of Sun Java System Directory Server.
- *Application Server* refers to an installed instance of Sun Java System Application Server (also known as Sun ONE Application Server.)
- *Web Server* refers to an installed instance of Sun Java System Web Server (also known as Sun ONE Web Server).

- *Web container that runs Identity Server* refers to the dedicated J2EE container (such as Web Server or Application Server) where the Policy and Management Services are installed.
- *IdentityServer_base* represents the base installation directory for Identity Server. The Identity Server 2004Q2 default base installation and product directories depend on your specific platform:
  - Solaris™ systems: `/opt/SUNWam`
  - Linux systems: `/opt/sun/identity`

  The product directory is `/SUNWam` for Solaris systems and `/identity` for Linux systems. When you install Identity Server 2004Q2, you can specify a different directory for `/opt` on Solaris systems or `/opt/sun` on Linux systems; however, do not change the `/SUNWam` or `/identity` product directory.

  For the base installation directory of the following products, refer to the documentation for the specific product.
- *DirectoryServer_base* represents the base installation directory for Sun Java System Directory Server.
- *ApplicationServer_base* is a variable place holder for the home directory for Sun Java System Application Server.
- *WebServer_base* is a variable place holder for the home directory for Sun Java System Web Server.

Related Information

Useful information can be found at the following locations:

- Directory Server documentation:

  http://docs.sun.com/coll/DirectoryServer_04q2
- Web Server documentation:

  http://docs.sun.com/coll/S1_websvr61_en
- Apllication Server documentation

  http://docs.sun.com/coll/s1_asseu3_en
- Web Proxy Server documentation:

  http://docs.sun.com/prod/s1.webproxys#hic

- Download Center:

  http://wwws.sun.com/software/download/

- Technical Support:

  http://www.sun.com/service/sunone/software/index.html

- Professional Services:

  http://www.sun.com/service/sunps/sunone/index.html

- Sun Enterprise Services, Solaris Patches, and Support:

  http://sunsolve.sun.com/

- Developer Information:

  http://developers.sun.com/prodtech/index.html

Related Third-Party Web Site References

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 by or in connection with the use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Chapter 1

Upgrading Identity Server 6.1 to Identity Server 2004Q2

This chapter describes how to upgrade Sun™ ONE Identity Server 6.1 or 6.1 Service Pack (SP) 1 to Sun Java™ System Identity Server 2004Q2. Topics include:

Requirements for Upgrading to Identity Server 2004Q2

The requirements for upgrading from Identity Server 6.1 to Identity Server 2004Q2 include:

Supported Platforms

Identity Server 2004Q2 is supported on these platforms:

- Solaris™ 9 Operating System (SPARC® Platform Edition)
- Solaris™ 9 Operating System (x86 Platform Edition)
- Solaris™ 8 Operating System (SPARC® Platform Edition)

For more information about these platforms, refer to the *Sun Java Enterprise System 2004Q2 Release Notes*.

Other platforms such as Windows are not supported in this release.

Identity Server Requirements

This chapter describes how to upgrade Identity Server 6.1 to Identity Server 2004Q2. To upgrade from Identity Server 6.0 or iPlanet™ Directory Server Access Management Edition (DSAME) 5.1, you must first upgrade the older version to Identity Server 6.1.

For information about upgrading older versions, see the instructions in the *Sun ONE Identity Server 6.1 Migration Guide* on the following web site:

`http://docs.sun.com/doc/816-6771-10`

The *Identity Server 6.1 Migration Guide* includes:

- Chapter 1, Upgrading from Identity Server 6.0 to Identity Server 6.1
- Chapter 2, Upgrading from DSAME 5.1 to Identity Server 6.0

Directory Server Requirements

Identity Server 2004Q2 supports either Directory Server 5 2004Q2 or Directory Server 5.1 Service Pack (SP) 1 (or newer).

If you want to upgrade Directory Server 5.1, follow the instructions in the *Directory Server 5 2004Q2 Installation and Migration Guide* on the following Web site:

`http://docs.sun.com/coll/DirectoryServer_04q2`

If you have Identity Server 6.1 or 6.1 SP1 and Identity Server 2004Q2 running concurrently against the same shared Directory Server, the Directory Server must be upgraded to include the Identity Server 2004Q2 schema elements. For other coexistence requirements, see "Identity Server Coexistence" on page 23.

Web Container Requirements

To upgrade to Identity Server 2004Q2, you must be using one of the following products as your web container:

- Web Server 6.1 SP2. To download this product, see:

  `http://www.sun.com/software/download/inter_ecom.html`

- Application Server 7.0 Update 3. To download this product, see:

  `http://www.sun.com/software/download/app_servers.html`

If you need to upgrade your web container, refer to "Upgrading the Web Container Software" on page 14.

Upgrading an Instance of Identity Server

This section includes the following information about upgrading an instance of Identity Server 6.1:

Backing up Web Container Customized Files

Before you upgrade, back up any web container customized files related to Identity Server 6.1, including:

- Customized console JSP pages
- Customized authentication JSP pages
- JAR files for authentication and customized modules

Tip: Make a list of your customizations so you can redo them after you upgrade and then verify that they work correctly.

Upgrading the Web Container Software

Identity Server 2004Q2 supports Web Server 6.1 SP2 or Application Server 7.0 Update 3 as a web container. If you are using an older version, you must upgrade the web container software before you can upgrade to Identity Server 2004Q2.

For information about upgrading web container software, refer to the respective web container documentation:

- For Web Server 6.1 SP2 see:

  `http://docs.sun.com/coll/S1_websvr61_en`

- For Application Server 7.0 Update 3, see:

  http://docs.sun.com/coll/s1_asseu3_en

Also, if you saved any customization files under "Backing up Web Container Customized Files" on page 14, you will need to redo the customizations after you upgrade the web container.

Running the Pre-Upgrade Script

The Identity Server 2004Q2 pre-upgrade script (`pre61to62upgrade`) is part of the Sun Java Enterprise System archive and is available in the following directory after you uncompress the archive:

JavaEnterpriseSystem_base`/Solaris_sparc/Product/identity_srv/Tools`

where *JavaEnterpriseSystem_base* is the directory where you uncompressed the archive.

The pre-upgrade script performs these functions:

- Backs up Identity Server 6.1 by running the `am2bak` script
- Removes the Identity Server 6.1 packages (but not Directory Server or web container packages) and then updates the `/var/sadm/install/productregistry` file to reflect that the packages have been removed
- Writes the `Sun_Java_System_Identity_Server_upgrade_log.`*timestamp* log file to the `/var/sadm/install/logs` directory

To run the pre-upgrade script, Directory Server must be running.

Before you run the pre-upgrade script, use the "Pre-Upgrade Script Worksheet" on page 76 to record the information you will need to provide.

To Run the Pre-Upgrade Script

1. Log in as or become superuser (`root`).
2. Verify that Directory Server is running. For example:

   ```
   # ps -ef | grep slapd
   ```

   If Directory Server is not running, start it. For example:

   ```
   # cd /var/opt/mps/serverroot/slapd-instance-name
   # ./start-slapd
   ```

3. Move to the directory where the pre-upgrade script exists and then run the script. For example:

```
# cd JavaEnterpriseSystem_base/Solaris_sparc/Product/identity_srv/Tools
# ./pre61to62upgrade
```

4. When you are prompted by the script, enter the following information:
   - Directory Server fully qualified host name. For example: ds.example.com
   - Directory Server port number. Default is 389.
   - Distinguished name (DN) and password of the top-level Identity Server administrator. For example: uid=amAdmin,ou=People,dc=example,dc=com
   - Directory where the script should back up the Identity Server 6.1 files. For example: /opt/is_backup
   - Certificate directory of the web container. For example: /opt/SUNWwbsvr/alias

The pre-upgrade script displays its status as it runs. Be sure to allow the script to finish completely. If you stop the script before it has finished, the results will be unpredictable.

After the script finishes, you are ready to install Identity Server 2004Q2.

Installing Identity Server 2004Q2

To install Identity Server 2004Q2, you must run the Sun Java Enterprise System installer. For information about the installer, refer to the *Sun Java Enterprise System Installation Guide* on the following web site:

http://docs.sun.com/coll/entsys_04q2

When you run the installer, you must provide the same information that was used for your Identity Server 6.1 (2003Q4) configuration, as described in this section.

Before you run the installer, use the "Identity Server 2004Q2 Installation Worksheets" on page 77 to record this information.

Identity Server 6.1 Information

- Administrator user ID (amadmin) and password
- LDAP user ID (amldapuser) and password
- Password encryption key (Multiple instances of Identity Server must use the same password encryption key value.)
- Services deployment URI
- Common domain deployment URI
- Console deployment URI
- Cookie domain

Web Container Used for Identity Server 6.1

- Web container: Web Server or Application Server
- Web container fully qualified host name
- Web container port number
- Web container installation directory path

Directory Server That Supported Identity Server 6.1

- Directory Server fully qualified host name
- Directory Server port number
- Directory Server root suffix
- Directory Manager DN and password

Other Installation Choices

Other installation choices you must make are:

- On the “Identity Server: Web container for running Identity Server services (4 of 6)” panel, for the Administration Console, check “Deploy new console”.
- On the “Identity Server: Directory Server Information (6 of 6)” panel, for “Is Directory Server provisioned with user data?”, check “yes” and provide values for the following marker and naming attributes:
  - Organization Marker Object Class (Default is SunISManagedOrganization)
  - Organization Naming Attribute (Default is o)
  - User Marker Object Class (Default is inetorgperson)
  - User Naming Attribute (Default is uid)

Running the Post-Upgrade Script

The Identity Server post-upgrade script (`Upgrade61DitTo62`) is available in the following directory after you install Identity Server 2004Q2:

- Solaris systems: *IdentityServer_base*`/SUNWam/migration/61to62/scripts`
- Linux systems: *IdentityServer_base*`/identity/migration/61to62/scripts`

where *IdentityServer_base* is the Identity Server 2004Q2 base installation directory. The default base installation directory is `/opt` on Solaris systems and `/opt/sun` on Linux systems.

The post-upgrade script performs these functions:

- Upgrades the Identity Server schema in the Directory Server to Identity Server 2004Q2
- Writes the `Sun_Java_System_Identity_Server_upgrade_dit_log.`*timestamp* log file to the `/var/sadm/install/logs` directory

To run the post-upgrade script, Directory Server must be running. During the script, you will be asked to restart Directory Server before the script can continue. At the end, you will also be asked to restart both Directory Server and the web container for the changes to take effect.

Before you run the pre-upgrade script, use the “Post-Upgrade Script Worksheet” on page 82 to record the information you will need to provide.

To Run the Post-Upgrade Script

1. Log in as or become superuser (`root`).
2. Verify that Directory Server is running. For example:

   ```
   # ps -ef | grep slapd
   ```

   If Directory Server is not running, start it. For example:

   ```
   cd /var/opt/mps/serverroot/slapd-instance-name
   ./start-slapd
   ```

3. Run the post-upgrade script. For example, on Solaris systems::

   ```
   cd IdentityServer_base/SUNWam/migration/61to62/scripts
   ./Upgrade61DitTo62
   ```

   where *IdentityServer_base* is the Identity Server 2004Q2 base installation directory.

4. When you are prompted by the script, provide the following information:
   - Directory Server fully qualified host name–For example: ds.`example.com`
   - Directory Server port number–Default is 389.
   - Distinguished name (DN) and password of the Directory Manager
   - Distinguished name (DN) and password of the top-level Identity Server administrator–For example: `uid=amAdmin,ou=People,dc=example,dc=com`
5. When you are prompted by the script, restart Directory Server. The script pauses for you to perform the restart.
6. After you restart Directory Server, return to the script and press Enter to continue. After the script has finished, it displays the following message:

   ```
   YOU MUST RESTART THE DIRECTORY AND WEB SERVERS FOR THE UPGRADE CHANGES
   TO TAKE EFFECT.
   ```

7. Restart Directory Server and the web container.

After Directory Server and the web container are running, you are ready to verify that the upgrade was successful.

Verifying the Upgrade

If you customized your Identity Server 6.1 installation, you must manually redo the customizations in your new Identity Server 2004Q2 installation.

Here are several ways to verify that the upgrade was successful:

- Access the Identity Server 2004Q2 console using the following URL:

  `http://`*host-name.domain-name*`:`*port*`/amconsole`

  where *host-name.domain-name*:*port* is the fully qualified host name and port of the web container you are using.

  When the Identity Server login page appears, log in as amadmin. Click the "Service Configuration" tab. If the new Identity Server 2004Q2 services such as "Discovery Service" and "Liberty and Personal Profile Service" are available, the upgrade of Identity Server on the specific web container should be successful.

- Review the status of the upgrade by checking the following log files in the `/var/sadm/install/logs` directory:

  - Pre-upgrade script (`pre61to62upgrade`):

    `Sun_Java_System_Identity_Server_upgrade_log.`*timestamp*

  - Sun Java Enterprise System installer:

    `Java_Shared_Component_Install.`*timestamp*
    `Java_Enterprise_System_install.A`*timestamp*
    `Java_Enterprise_System_install.B`*timestamp*
    `Java_Enterprise_System_Summary_Report_install.`*timestamp*

  - Post-upgrade script (`Upgrade61DitTo62`):

    `Sun_Java_System_Identity_Server_upgrade_dit_log.`*timestamp*

Upgrading Multiple Instances

This section describes how to upgrade multiple Identity Server 6.1 instances running on different hosts that share the same Directory Server.

NOTE The upgrade process supports multiple instances of Identity Server installed on different host systems. Upgrading multiple instances of Identity Server installed on the same host system is not supported in the current release. If you have multiple instances on the same host, after you upgrade the main instance, you must then recreate the additional instances.

Identity Server 6.1 and Identity Server 2004Q2 instances installed on different hosts can run concurrently against the same shared Directory Server. For more information, including the Directory Server requirements, see "Identity Server Coexistence" on page 23.

To Upgrade an Instance

1. Log in as or become superuser (`root`).
2. Stop all Identity Server 6.1 instances that access the Directory Server. For example, on Solaris systems:

   ```
   # cd /IdentityServer_base/SUNWam/bin
   # ./amserver stop
   ```

   where *IdentityServer_base* is the Identity Server 6.1 base installation directory.

   Stopping all instances prevents Identity Server from making changes to the Directory Server while you are performing the upgrade.
3. Start the Identity Server 6.1 instance you want to upgrade. For example:

   ```
   # ./amserver start
   ```
4. Upgrade the Identity Server 6.1 instance you started in Step 3, as described in "Upgrading an Instance of Identity Server" on page 14.

   During the upgrade of the first instance, the post-upgrade script (`Upgrade61DitTo62`) upgrades the Identity Server schema to Identity Server 2004Q2. During subsequent upgrades of other instances, however, the post-upgrade script detects that the Directory Server has already been upgraded and does not try to upgrade it again.

5. Restart the instance you just upgraded.

   Repeat Step 3 through Step 5 for each Identity Server 6.1 instance on a different host that you want to upgrade.

6. If there are any Identity Server 6.1 instances you did not upgrade, restart those instances. For information about the co-existence of Identity Server 6.1 and Identity Server 2004Q2, see "Identity Server Coexistence" on page 23.

Upgrading the Identity Server SDK

To upgrade an Identity Server 2003Q4 (6.1) SDK only installation, you must uninstall the 2003Q4 version and then re-install the 2004Q2 version.

To upgrade an Identity Server SDK only installation

1. Back up your Identity Server 2003Q4 configuration files, including the `AMConfig.properties` and `serverconfig.xml` files. (The upgrade process will not affect your user data.)

2. Uninstall the Identity Server 2003Q4 SDK by following the instructions in the *Sun Java Enterprise System 2003Q4 Installation Guide* (`http://docs.sun.com/doc/816-6874`).

3. Install the Identity Server 2004Q2 SDK by following the instructions in the *Sun Java Enterprise System 2004Q2 Installation Guide* (`http://docs.sun.com/doc/817-5760`).

4. Incorporate the configuration changes you saved in Step 1 into the new Identity Server 2004Q2 configuration files.

Identity Server Coexistence

Identity Server 6.1 and Identity Server 2004Q2 can coexist and run concurrently against the same shared Directory Server, if these requirements are met:

- The Identity Server 6.1 version can either be 6.1 or 6.1 SP1.
- Identity Server 6.1 or 6.1 SP1 and Identity Server 2004Q2 must be installed on different hosts.
- The Directory Server version must be 5.2.
- The Directory Server schema must be upgraded to include the Identity Server 2004Q2 schema elements.

Usually, the coexistence of Identity Server 6.1 and Identity Server 2004Q2 is a transitional phase during an Identity Server 2004Q2 upgrade. During the upgrade process, some Identity Server 6.1 servers are upgraded to version 2004Q2 before the other version 6.1 servers are upgraded. The Directory Server is upgraded to the version 2004Q2 schema when you upgrade the first Identity Server 6.1 server.

Then, both any upgraded version 2004Q2 servers and any remaining Identity Server 6.1 servers and applications can run against the upgraded Directory Server.

To access the Identity Server 2004Q2 features, including new services, new attributes in existing services, and new policy plug-ins, use the Identity Server 2004Q2 console. Do not use the Identity Server 6.1 admin console to access Identity Server 2004Q2.

Using Portal Server Mobile Access

To use Java System Portal Server Mobile Access, change the Identity Server Client Detection global attributes as follows:

1. Access the Identity Server 2004Q2 console using the following URL:

   `http://`*host-name.domain-name*`:`*port*`/amconsole`

   where *host-name.domain-name*:*port* is the fully qualified host name and port of the web container you are using.

2. When the Identity Server login page appears, log in as `amadmin`.

3. On the console, click the Service Configuration tab.

   The console displays the Service Configuration options in the navigation frame.

4. In the navigation frame under Service Configuration, click Client Detection.

5. For Client Detection, set the following items in the data frame:

   a. Set the Client Detection Class global attribute to `com.sun.mobile.cdm.FEDIClientDetector`

   b. Click the Enable Client Detection check box.

6. Click Save.

Chapter 2

Configuring Identity Server with a Provisioned Directory

This chapter describes how to install and configure Sun Java™ System Identity Server 2004Q2 with a directory that is already provisioned with user data, including how to configure Identity Server to work with your directory information tree (DIT) and how to make the necessary changes to your existing Directory Server and directory entries. The number and scope of changes you will need to make depends on how your directory tree is structured and how you plan to use Identity Server.

This chapter includes the MadisonParc example, which is described in "MadisonParc Examples Used in This Chapter" on page 27.

Topics in this chapter include:

Overview of Installation and Configuration Tasks

This section describes the steps you must perform before you will see your existing directory data in Identity Server. This chapter groups these steps into two distinct phases: installation and post-installation configuration.

Installing and Starting Identity Server 2004Q2

1. Install Identity Server 2004Q2.

   To install Identity Server 2004Q2, you must run the Sun Java Enterprise System installer. When the installer asks “Is Directory Server provisioned with user data?”, answer “yes”. For information about the installer, refer to the *Sun Java Enterprise System 2004Q2 Installation Guide* on the following documentation web site:

   `http://docs.sun.com/coll/entsys_04q2`

2. Start Identity Server.

   For detailed instructions, see “Starting Identity Server and Logging In” on page 30 in this chapter.

Post-Installation Configuration

Post-installation configuration includes these steps:

1. Manually configure Directory Server to work with Identity Server.

   In this step you enable the Directory Server referential integrity plug-in and create new database indexes. See “Configuring Directory Server” on page 35 in this chapter.

2. Enable User Management services, as follows:

   a. Add Marker Object Classes to Existing Directory Entries.

      Before Identity Server can recognize the data in an existing directory, you must add special object classes to entries for all organizations, groups and users that will be managed by Identity Server. Sample scripts are bundled in the product to help you automatically add these object classes to your directory. Detailed steps and examples are provided in this chapter. See “Adding Identity Server Object Classes to Existing Directory Entries” on page 39.

b. Add Custom Object Classes to Identity Server Schema.

If your existing directory tree contains custom object classes, Identity Server won't recognize them until you manually configure both Identity Server and Directory Server. The types of changes you need to make are illustrated in this chapter using the directory tree for MadisonParc, a fictitious company. Detailed steps are in "Adding Custom Object Classes to Identity Server Schema" on page 59.

c. Load Identity Server LDIF into your directory.

In this step, you commit all of your LDIF modifications to make actual changes in the Directory Server. Detailed instructions are in "Loading Identity Server LDIF into Your Directory" on page 71.

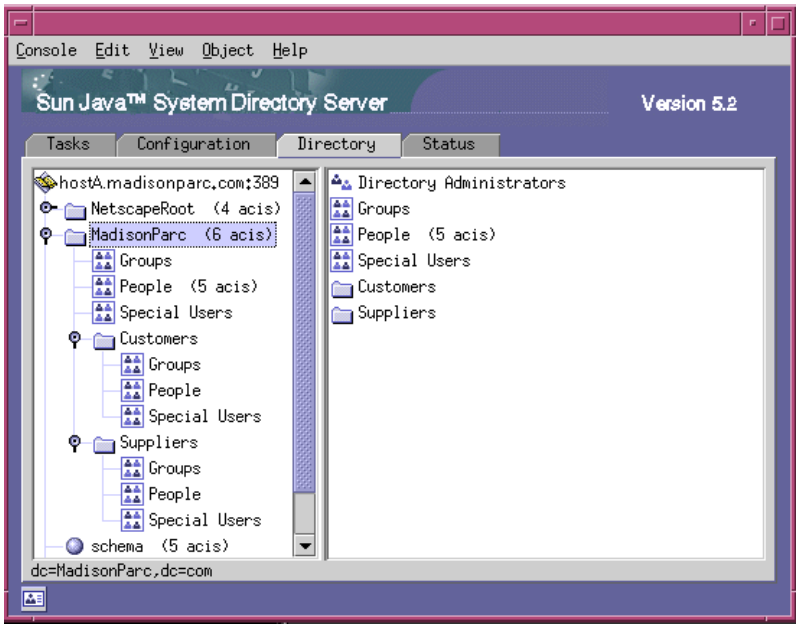
d. In the Identity Server Console, enable User Management.

This is the last step in the instructions for "Enabling User Management Services" on page 37.

MadisonParc Examples Used in This Chapter

The MadisonParc examples in this chapter show the types of manual modifications you must make in both Directory Server and Identity Server. Figure 2-1 illustrates the Directory Server console view of the directory tree for MadisonParc. The tree includes three organizational units (ou) at the top level of the tree: Groups, People, and Special Users. These organizational units contain entries for MadisonParc employees. Two organizations (dc), Customers and Suppliers, were created under the root level to contain entries for non-employees.

Figure 2-1 Existing Directory Tree for MadisonParc and Identity Server



The MadisonParc example has two custom object classes and three custom attributes. These object classes and attributes are not included in the Identity Server schema nor in the Directory Server 5.1 SP1 schema.Table 2-1 summarizes the custom objects and their uses in the MadisonParc directory tree.

Table 2-1 User-defined Objects Used in the MadisonParc Directory Tree

Object	Description
`madisonparc-org`	Object class added to all organization entries.
`madisonparc-org-description`	Attribute added to each organization entry; required by `madisonparc-org`.
`company`	Object class added to all user entries.
`acctNumber`	Attribute added to each user entry; required by the `company` object class.
`companyName`	Attribute added to each user entry; required by the `company` object class.

Before a MadisonParc administrator can use Identity Server to manage these extensions, you must make the following modifications to the Identity Server schema:

- Add the two custom object classes and three custom attributes to `umsExisting.xml`.
- Add `madisonparc-org` to `amEntrySpecific.xml`.
- Add `madisonparc-org-description` to `amEntrySpecific.properties`.
- Add `companyName` and `acctNumber` to `amUser.xml`.
- Add `companyname` and `acctNumber` to `amUser.properties`.

If your existing directory tree contains custom object classes or attributes, you'll need to make similar changes in your directory and in your Identity Server XML files.

Detailed steps and examples are provided in this chapter. See "Adding Custom Object Classes to Identity Server Schema" on page 59.

Directory Server Considerations

Before you install Identity Server against an existing Directory Server that is provisioned with user data, consider the following Directory Server issues:

- "Migrating Pre-2004Q2 Versions of Directory Server" on page 29
- "Backing Up Your Directory Server" on page 30
- "Accessing Directory Server" on page 30

For more information, see the documentation for Directory Server on the following Web site:

`http://docs.sun.com/coll/DirectoryServer_04q2`

Migrating Pre-2004Q2 Versions of Directory Server

If you are using a pre-2004Q2 version of Directory Server, you can upgrade your existing Directory Server to version 2004Q2 and then migrate your existing data to the upgraded directory. For detailed instructions, see the *Directory Server 5 2004Q2 Installation and Migration Guide.*

Backing Up Your Directory Server

The installation and post-installation tasks described in this chapter make changes to your existing directory. Before you begin, back up your Directory Server using `db2ldif`, `db2bak`, or the Directory Server Console. For information about backing up your directory, see the *Sun ONE Directory Server 5.2 Administration Guide.*

Accessing Directory Server

To access Directory Server, you must also have the appropriate administrator privileges to modify user entries.

Installing Identity Server User and Policy Management Services

See the *Java Enterprise System Installation Guide* for instructions about installing Identity Server User and Policy Management Services. The Java Enterprise System installer will guide you to provide information about your existing provisioned directory. After the installation program is finished, you can start Identity Server and log in to its administration console.

Starting Identity Server and Logging In

After you've installed Identity Server and configured Directory Server appropriately, you can check the installation. Start Identity Server and log in to the Identity Server Console as the user `amAdmin`. After you successfully log in, you'll see the Identity Server web interface.

To Start Identity Server

1. Start Directory Server.

   Execute the `start` or `restart` command in the directory where the Directory Server instance is installed. For example, on Solaris systems:

   DirectoryServer_base`/slapd-`*hostName*`/slapd-start`

   or:

   DirectoryServer_base`/slapd-hostName/slapd-restart`

2. Start the web container that Identity Server is using.

   If the web container is Web Server, execute the `start` command in directory were the Web Server instance is installed. For example, on Solaris systems:

   WebServer_base`/https-`*hostName*`/start`

   If the web container is Application server, execute the `start` command in the directory where the Application Server instance is installed. For example, on Solaris systems:

   ApplicationServer_base`/bin/asadmin start-`*server1* `--user` *admin_user*`--password`

   where *server1* is the instance name by default

   If the web container is BEA WebLogic or IBM WebSphere, refer to the instructions for starting the server in the documentation that comes with the product.

To Log in to the Identity Server Console

1. Access the login URL as follows:

   `http://`*host.domain*`:`*port*`/amserver/`

   where *host-name.domain-name*:*port* is the fully qualified host name and port of the web container you are using. For example:

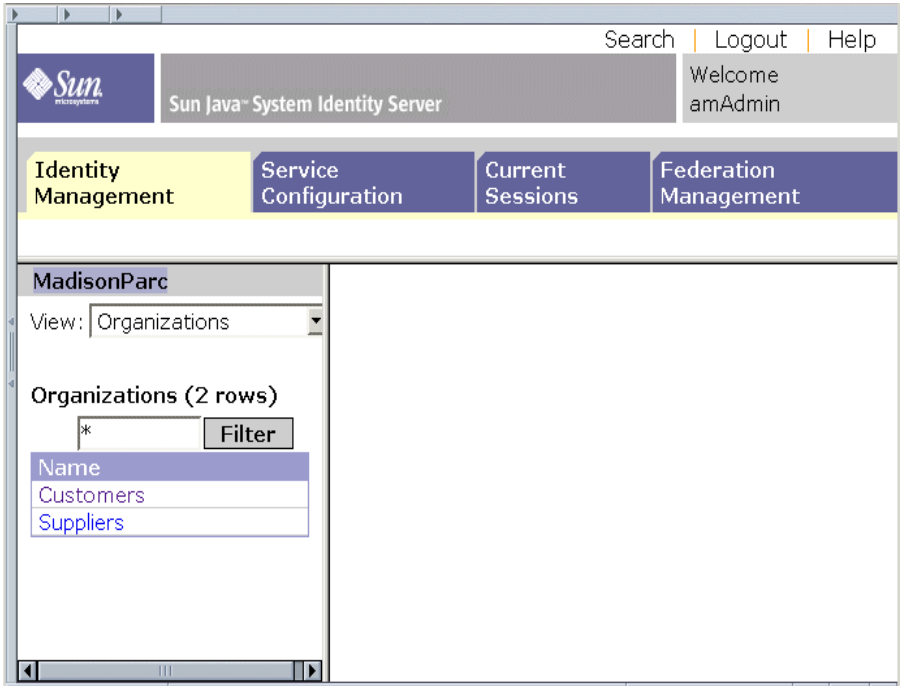
   `http://ishost.sesta.com:58080/amserver/`

2. On the login page, enter the Top-Level Administrator user name `amAdmin` and the password you specified during installation.

The Policy Management services are automatically installed against your existing provisioned directory. When you see the Identity Server interface (see Figure 2-2), you can immediately begin creating policies. For more information, see the *Java System Identity Server 2004Q2 Administration Guide*.

You will see the root suffix and organizations you specified during installation. For the MadisonParc example used at the beginning of this chapter, you would see the root suffix MadisonParc, and the two organizations `Customers` and `Suppliers`.

Figure 2-2 First-time Login for MadisonParc

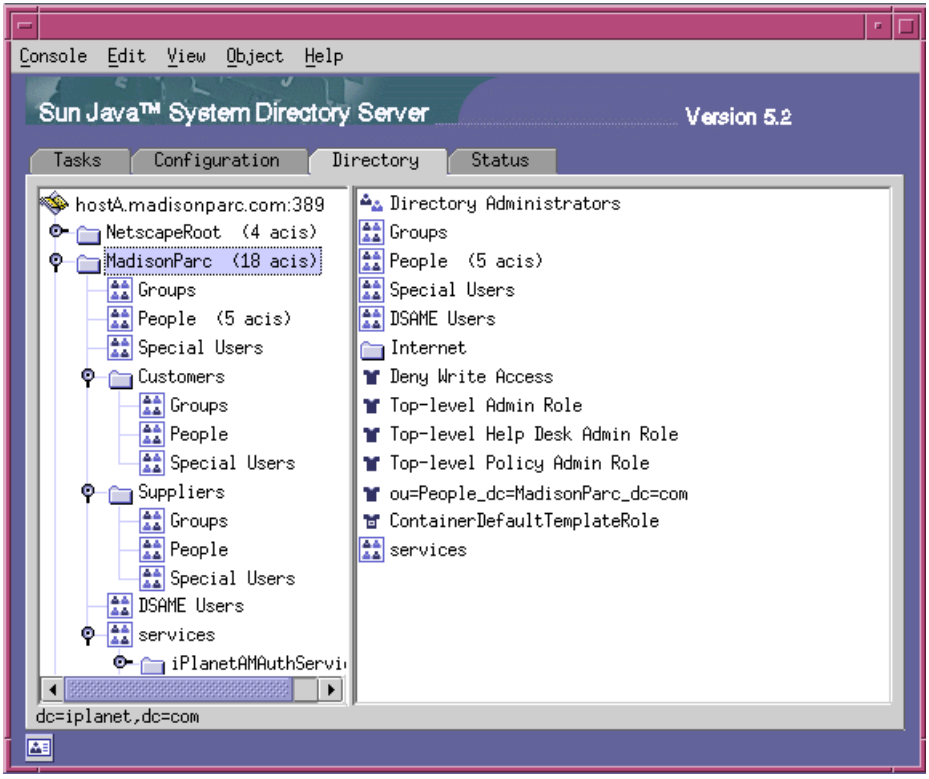


NOTE You will not see directory entries below the organization level until you complete the post-configuration tasks described in the following sections:

- "Configuring Directory Server" on page 35
- "Enabling User Management Services" on page 37

Also immediately after Installation, if you look in the Directory Server console view, you'll see the Identity Server object classes, roles, users, and services Figure 2-3 illustrates the Identity Server objects and existing directory tree for the MadisonParc examples used in this chapter.

Figure 2-3 Policy Management Services Installed Against the Existing MadisonParc Directory Tree



Configuring Directory Server

After you've installed the Identity Server schema, you must configure Directory Server to work with Identity Server. Perform the steps in the following procedures:

- Enable the Directory Server referential integrity plug-in
- Add Identity Server indexes

When the referential integrity plug-in is enabled, it performs integrity updates on specified attributes immediately after a delete or rename operation. This ensures that relationships between related entries are maintained throughout the database. Database indexes enhance the search performance in Directory Server.

NOTE Before continuing with these procedures, be sure that Directory Server is running.

To Enable the Referential Integrity Plug-In

1. In Directory Server console, click Configuration.
2. In the navigation tree, double-click Plug-ins to expand the list of Plug-ins.
3. In the Plug-ins list, click "referential integrity postoperation."
4. In the properties area, check the "Enable plug-in" box.
5. Click Save.

The plug-in is not enabled until you restart Directory Server.

To Add Identity Server Indexes

1. In Directory Server console, click Configuration.
2. Add the `nsroledn` index:
   a. In the navigation tree, double-click the Data icon, and then click the root suffix that contains the directory entries you want to use in Identity Server.
   b. Click the Indexes tab.
   c. Under "Additional Indexes," for the `nsroledn` attribute, check the following check boxes: Equality, Presence, and Substring.
   d. Click Save.

e. In the “Indexes” window, after the index is successfully created, click Close.

3. Add the `memberof` index:

   a. In the Indexes tab, click “Add attribute...”

   b. In the “Select Attributes” window, select the attribute `memberof`, and then click OK.

   c. In the Indexes tab, for the `memberof` attribute, check the following check boxes: Equality and Presence.

   d. Click Save.

   e. In the “Indexes” window, after the index is successfully created, click Close.

4. Add the `iplanet-am-static-group` index:

   a. In the Indexes tab, click “Add attribute...”

   b. In the “Select Attributes” window, select the attribute `iplanet-am-static-group`, and then click OK.

   c. In the Indexes tab, for the `iplanet-am-static-group` attribute, check the following Equality check box.

   d. Click Save.

   e. In the “Indexes” window, after the index is successfully created, click Close.

5. Add the `iplanet-am-modifiable-by` index:

   a. In the Indexes tab, click “Add attribute...”

   b. In the “Select Attributes” window, select the attribute `iplanet-am-modifiable-by`, and then click OK.

   c. In the Indexes tab, for the `iplanet-am-modifiable-by` attribute, check the following check box: Equality.

   d. Click Save.

   e. In the “Indexes” window, after the index is successfully created, click Close.

6. Add the `iplanet-am-user-federation-info-key` index:

   a. In the Indexes tab, click “Add attribute...”

b. In the “Select Attributes” window, select the attribute `iplanet-am-user-federation-info-key`, and then click OK.

c. In the Indexes tab, for the `iplanet-am-user-federation-info-key` attribute, check the following check box: Equality.

d. Click Save.

e. In the “Indexes” window, after the index is successfully created, click Close.

7. Restart Directory Server.

Enabling User Management Services

This section provides an overview of the configuration tasks you’ll need to perform in order to see your existing user data displayed in the Identity Server interface. If you want to enable user management before proceeding with the configuration tasks, first log in to Identity Server, and then skip to step Step 8 this section. Note that if you skip to step Step 8 without performing the necessary configuration, although you will be able to create and manage new user entries, you will not be able to access your existing data through Identity Server.

To enable User Management Services

1. Add Identity Server object classes and attributes to your existing DIT.

   For detailed instructions, see “Adding Identity Server Object Classes to Existing Directory Entries” on page 39 in this chapter.

2. Modify the Identity Server schema.

   See “Adding Custom Object Classes to Identity Server Schema” on page 59 in this chapter for more information. The section provides detailed instructions on performing the following steps:

   a. Modifying the Creation Templates

   b. Adding Attributes to the Organization Schema

   c. Adding Attributes to the User Schema

3. To load all XML files, enter the following command:

   ```
   /etc/opt/SUNWam/config/ums/amserveradmin
     "user_naming_attibute=amadmin,ou=people,root_suffix"  password
   ```

4. Load `installExisting.ldif` file into your Directory Server.

   For detailed instructions, see "Loading Identity Server LDIF into Your Directory" on page 71 in this chapter.

5. Load `install.ldif` file into your Directory Server.

   For detailed instructions, see "install.ldif" on page 73 in this chapter.

6. Restart Identity Server. To start Identity Server services, you must start both Directory Server and the web container that runs Identity Server.

   a. Start Directory Server.

      Execute the `start` or `restart` command in the directory where the Directory Server instance is installed. For example, on Solaris systems:

      DirectoryServer_base`/slapd-`*instanceName*`/slapd-start`

   b. Start the web container that runs Identity Server.

      - If the web container is Web Server, then execute the `start` command in directory were the Web Server instance is installed. For example, on Solaris systems:

        WebServer_base`/https-`*instanceName*`/start`

      - If the web container is Application Server, then execute the `start` command in the directory where the Application Server instance is installed. For example, on Solaris systems, where *server1* is the instance name by default:

        ApplicationServer_base`/bin/asadmin start-`*server1*
        `--user` *admin_user* `--password` *your_admin_passwd server1*

      - If the web container is BEA WebLogic or IBM WebSphere, then see the instructions for starting the server in the documentation that comes with the product.

7. Log in to Identity Server Console as `amAdmin`.

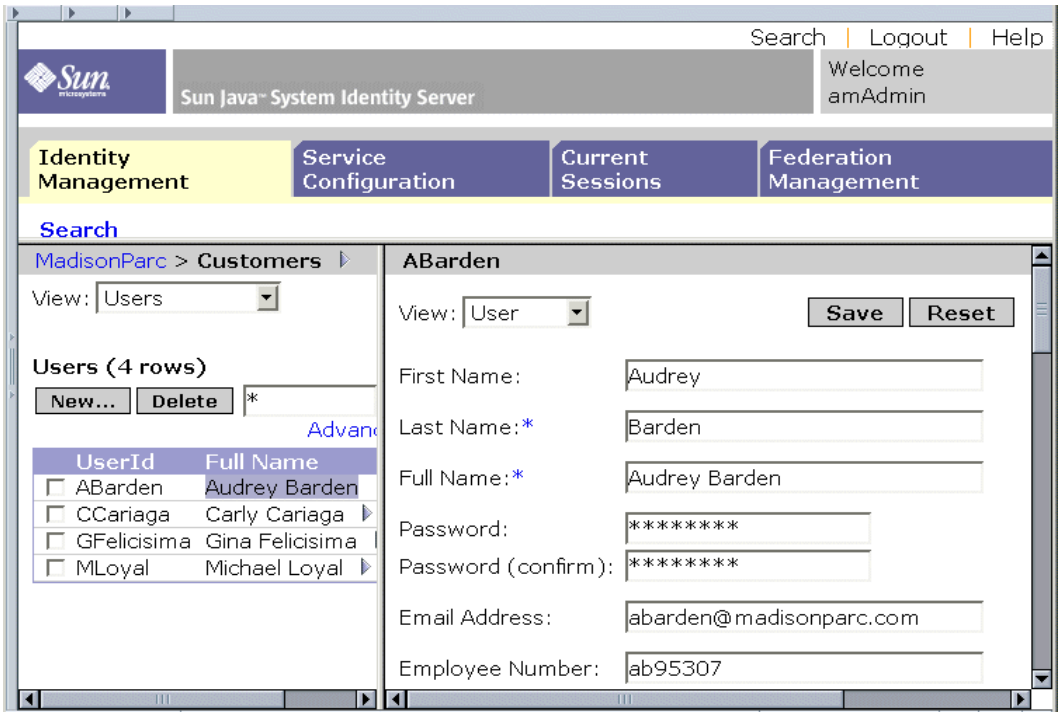
   You will not see your existing groups and users from your existing directory tree until you complete the following two steps.

8. In the Identity Server Console, click Service Management and then Administration.

9. In the Administration window, click the "Enable User Management" box and then click Save.

Once you've checked the "Enable User Management" box, you should see all entries beneath the organization level in Identity Server. For instructions on using Identity Server to create or manage users and policies, see the *Java System Identity Server 2004Q2 Administration Guide*.

Figure 2-4 Identity Server With Data From Existing MadisonParc Directory Tree



Adding Identity Server Object Classes to Existing Directory Entries

After you've installed configured Identity Server, you must modify your existing directory entries to include the necessary Identity Server object classes and attributes. You can think of the Identity Server object classes as *markers* that indicate the directory entries you want to manage through Identity Server. These markers enable Identity Server to recognize the entries in your directory. The object classes contain special attributes that are necessary to achieve delegated administration.

Before You Begin

There are a number of resources you can use to facilitate the remaining steps for using an existing directory.

Examples Used in This Section

The examples used in this chapter are based on the directory tree for a fictitious company named MadisonParc. Figure 2-5 shows two organizations, `Customers` and `Suppliers`, under the root.

Utilities and Scripts You Can Use

You can make these modifications by using Directory Server Console, or by using the `ldapmodify` or `db2ldif` utilities that come with Directory Server. You can also use the sample scripts that come with Identity Server.

Directory Server Utilities

Make sure that you're using the appropriate version of `ldapmodify`. Set your path to use the `ldapmodify` command that is included with Directory Server. (Do not use the version included with Solaris, which is in the `/bin` or `/usr/bin` directory.)

On Solaris systems, add *IdentityServer_base*`/SUNWam/ldaplib/ldapsdk` to your `LD_LIBRARY_PATH` to access the appropriate Directory Server libraries.

At the command line, enter:

```
which ldapmodify
```

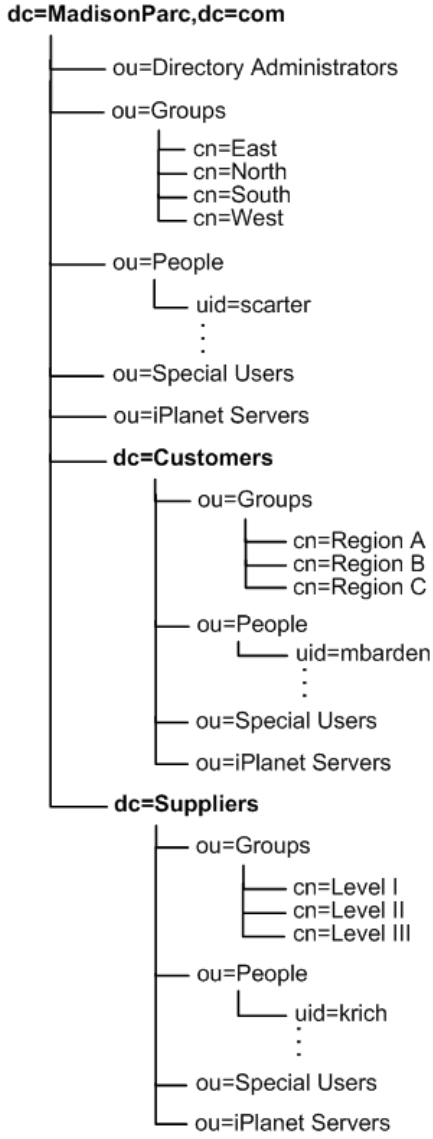
The following should be displayed on Solaris systems:

IdentityServer_base`/SUNWam/bin/ldapmodify`

For detailed information about how to make changes to the directory using these utilities or by using the console, see the documentation for Directory Server:

`http://docs.sun.com/coll/DirectoryServer_04q2`

Figure 2-5 Existing MadisonParc Directory Tree



Sample Migration Scripts

The sample scripts included with Identity Server require Perl 5.x or later. Table 2-2 provides descriptions of what each script adds to existing directory entries. You'll find the sample scripts in the following location on Solaris systems:

- Solaris systems: *IdentityServer_base*/SUNWam/migration
- Linux systems: *IdentityServer_base*/identity/migration

Table 2-2 Scripts for Adding Identity Server Marker Object Classes

Script	What it Does
update-o.pl	Adds the following to each organization entry: • sunManagedOrganization • sunISManagedOrganization • sunNameSpace • inetDomain • inetDomainStatus
update-people.pl	Adds iplanet-am-managed-people-container to each people container.
update-ou.pl	Adds iplanet-am-managed-org-unit to each organizational unit.
update-users.pl	Adds the following to each user entry: • inetadmin • iplanet-am-managed-person • iplanet-am-user-service • inetuser • iPlanetPreferences • inetOrgPerson
udpate-static-groups.pl	Adds the following to each static group: • iplanet-am-managed-static-group • iplanet-am-managed-group
update-filtered-groups	Adds the following to each dynamic, or *filtered*, group: • iplanet-am-managed-group • iplanet-am-managed-filtered-group

Table 2-2 Scripts for Adding Identity Server Marker Object Classes *(Continued)*

Script	What it Does
`update-assignable-dynamic-groups`	Adds the following to each assignable dynamic group: • `iplanet-am-managed-group` • `iplanet-am-managed-assignable group`
`update-groups.pl`	Adds `iplanet-am-managed-group-container` to each organizational unit that contains groups.

While these samples should prove useful, keep in mind that they are only tools to assist you in properly formatting the directory tree and other data. Each script generates an LDIF file that you can inspect before making actual changes in your directory. You run each script a second time with the last line uncommented to make the actual changes. Steps for using each sample script are included in this chapter under the following headings:

- Marking Organizations
- Marking People Containers
- Marking Organizational Units
- Marking Users
- Marking Static Groups
- Marking Dynamic (Filtered) Groups
- Marking Assignable Dynamic Groups
- Marking Group Containers

Important: The changes made by using these scripts cannot be automatically undone. Be sure to back up your data before running each script.

Two Approaches to Modifying the Existing Directory Tree

You can use one of two approaches for modifying the directory tree. One option is to make all the necessary modifications to your directory tree before loading the Identity Server LDIF and XML configuration files. This procedure is more error-prone, but may be faster if you have experience using LDAP.

The other option is to make a few modifications in your LDIF and XML files, and then start Identity Server to make sure those modifications were done correctly. This second approach is the recommended approach. For example, you may want to add the Identity Server object classes for each of your organizations, restart Identity Server, and verify that your organizations appear in the Identity Server Administration Console. Then add marker object classes for groups, check them and so forth.

Marking Organizations

If you used an existing organization as your default organization during installation, you do not have to make these changes. The installation program automatically added these object classes and attributes. Skip to "Marking People Containers" on page 46.

If you have sub-organizations or custom organizations you must make the following changes:

1. Add the following object classes to each organization entry:
   - `sunManagedOrganization`
   - `sunNameSpace`
   - `inetDomain`
2. Add the following attribute to each organization entry:
   - `inetDomainStatus`

In the MadisonParc example, these object classes and their attributes are added to the organizations `dc=Customers` and `dc=Suppliers`.

To Mark Organizations Using the Sample Script

1. Copy `update-o.pl` to the following directory:

   DirectoryServer_base`/shared/bin`

2. Set the `$base` variable to the base suffix of the directory tree to be managed by Identity Server. Example: `dc=MadisonParc,dc=com`

3. In the directory where the script is located, enter the following command:

   ```
   perl update-o.pl
   ```

4. When prompted, provide the following information:

   Enter Host Name: Enter the name of the computer system in which your Directory Server is installed.

   Enter Bind User Name: Enter a user name that has sufficient privileges for accessing the entire directory. Example: `cn=Directory Manager`

   Enter Bind password: Enter the password for the user you specified above.

   Enter port number: Enter the Directory Server port number. Example: `389`

5. Check the results in the file `orgs-updated.ldif` that is generated by the script, and verify that the appropriate changes are listed. The changes contained in this file will automatically be made in the directory in the next step.

   Important Note: If there are organizations that you do not want to be managed by Identity Server, you should delete those entries from this `orgs-updated.ldif` now. Then, instead of going on to the next step, manually load the file using the following command:

   ```
   ldapmodify -h hostname -p port -D bind_user -w password -a -c -f
       orgs-updated.ldif
   ```

6. In the script `update-o.pl`, uncomment the last line and replace variables appropriately. For example, to add marker object classes to MadisonParc directory entries, the last line of the script is changed from this:

   ```
   #system("$LDAP_MODIFY -h'$host' -p'$port' -D'$bind_user'
       -w'$bind_pwd' -a -c -f orgs-updated.ldif");
   ```

   to this:

   ```
   system("$LDAP_MODIFY -h'ginac.MadisonParc.com' -p'389'
       -D'cn=Directory Manager' -w'password' -a -c -f
        orgs-updated.ldif");
   ```

In Code Example 2-1, the modifications to the MadisonParc directory entries are indicated in bold:

Code Example 2-1 Organization Entries With Marker Object Classes

```
...
dn: dc=Customers,dc=MadisonParc,dc=com
dc: Customers
objectClass: top
objectClass: domain
objectClass: external
objectClass: sunManagedOrganization
objectClass: sunNameSpace
objectClass: inetDomain
inetDomainStatus: Active


dn: dc=Suppliers,dc=MadisonParc,dc=com
dc: Suppliers
objectClass: top
objectClass: domain
objectClass: external
objectClass: sunManagedOrganization
objectClass: sunNameSpace
objectClass: inetDomain
inetDomainStatus: Active
...
```

Marking People Containers

People containers are typically assigned the ou attribute and are used to store all user entries for a branch of the directory. To each people container, add the `iplanet-am-managed-people-container` object class.

To Mark People Containers Using the Sample Script

1. Copy `update-people.pl` to the following directory:

   DirectoryServer_base`/shared/bin`

2. Be sure the `$base` variable is set to the base suffix of the directory tree to be managed by Identity Server. For example: `dc=MadisonParc,dc=com`

   In the MadisonParc example, the script was also modified to include people containers located under the organizations. In Code Example 2-2, bold indicates the change in the search scope.

Code Example 2-2 Scope in `update-people-container.pl` is Modified

```
# run search to find all people containers, putting their DNs in to a
file
system("$LDAP_SEARCH -h \"$host\" -p \"$port\" -D \"$bind_user\"
    -w \"$bind_pwd\" -b \"$base\" -s sub -T \"(&(ou=$people)
       (!(objectclass=iplanet-am-*)))\" dn > people.dn");
```

3. In the directory where the script is located, at the command line enter the following:

   `perl update-people.pl`

4. When prompted, provide the following information:

   Enter Host Name: Enter the name of the computer system in which your Directory Server is installed.

   Enter Bind User Name: Enter a user name that has sufficient privileges for accessing the entire directory. Example: `cn=Directory Manager`

   Enter Bind password: Enter the password for the user you specified above.

   Enter port number: Enter the Directory Server port number. For example: `389`

   Enter People Container: Enter the name of the people container that contains the uids you want to modify. For example: `People`

5. Check the results in the file `people-updated.ldif` which is created in the same directory as the script, and verify that the appropriate changes were made. The changes contained in this file will automatically be made in the directory in the next step.

   Important Note: If there are people containers that you do not want to be managed by Identity Server, you should delete those entries from `people-updated.ldif` now. Then, instead of going on the to next step, manually load the file using the following command:

   `ldapmodify -h` *hostname* `-p` *port* `-D` *bind_user* `-w` *password* `-a -c -f people-updated.ldif`

6. In the script `update-people.pl`, uncomment the last line and replace variables appropriately. In the MadisonParc example, the last line of the script is changed from this:

   ```
   #system("$LDAP_MODIFY -h'$host' -p'$port' -D'$bind_user'
     -w'$bind_pwd' -a -c -f people-updated.ldif");
   ```

   to this:

   ```
   system("$LDAP_MODIFY -h'ginac.MadisonParc.com' -p'389'
     -D'cn=Directory Manager' -w'password' -a -c -f
     people-updated.ldif");
   ```

   In Code Example 2-3, marker object class for the people container under dc=Customers is indicated in bold.

Code Example 2-3 People container entry with marker object class.

```
...
dn: ou=People,dc=Customers,dc=MadisonParc,dc=com
ou: People
objectClass: top
objectClass: organizationalunit
objectClass: iplanet-am-managed-people-container
```

Marking Organizational Units

Organizational units are typically assigned the ou attribute. To each container that is an organizational unit, add the following object class:

`iplanet-am-managed-org-unit`

To Mark Organizational Units Using the Sample Script

1. Copy `update-ou.pl` to the following directory:

   DirectoryServer_base`/shared/bin`

2. Set the `$base` variable to the base suffix of the directory tree to be managed by Identity Server. For example: `dc=MadisonParc,dc=com`.

3. In the directory where the script is located, at the command line enter the following:

   ```
   perl update-ou.pl
   ```

4. When prompted, provide the following information:

   Enter Host Name: Enter the name of the computer system in which your Directory Server is installed.

   Enter Bind User Name: Enter a user name that has sufficient privileges for accessing the entire directory. For example: `cn=Directory Manager`

   Enter Bind password: Enter the password for the user you specified above.

   Enter port number: Enter the Directory Server port number. For example: `389`

5. Check the results in the file `orgunit-updated.ldif` which is created in the same directory as the script, and verify that the appropriate changes are listed. The changes contained in this file will automatically be made in the directory in the next step.

   Important Note: If there are organizational units that you do not want to be managed by Identity Server, you should delete those entries from this `ou-updated.ldif` now. Then, instead of going on the to next step, manually load the file using the following command:

   ```
   ldapmodify -h hostname -p port -D bind_user -w password -a -c -f
     orgunit-updated.ldif
   ```

6. In the script `update-ou.pl`, uncomment the last line and replace variables appropriately. In the MadisonParc example, the last line of the script is changed from this:

   ```
   #system("$LDAP_MODIFY -h'$host' -p'$port' -D'$bind_user'
     -w'$bind_pwd' -a -c -f orgunit-updated.ldif");
   ```

   to this:

   ```
   system("$LDAP_MODIFY -h'ginac.MadisonParc.com' -p'389'
     -D'cn=Directory Manager' -w'password' -a -c -f
     orgunit-updated.ldif");
   ```

In Code Example 2-4, marker object class for the organizational units under dc=MadisonParc,dc=com is indicated in bold.

Code Example 2-4 Organizational Unit Entry With Marker Object Class

```
...
dn: ou=People,dc=Customers,dc=MadisonParc,dc=com
ou: People
objectClass: top
objectClass: organizationalunit
objectClass: iplanet-am-managed-people-container
...
```

Marking Users

To each user entry, add the following object classes:

- iplanet-am-managed-person
- iplanet-am-user-service
- inetuser
- iPlanetPreferences
- inetOrgPerson
- inetadmin

To Mark Users Using the Sample Script

1. Copy update-users.pl to the following directory:

   DirectoryServer_base/shared/bin

2. Be sure the $base variable is set to the base suffix of the directory tree to be managed by Identity Server. For example: dc=MadisonParc,dc=com

3. Be sure the $base-component variable is set to the base suffix of the directory tree.For example: dc=MadisonParc,dc=com

4. In the directory where the script is located, at the command line enter the following:

   perl udpate-users.pl

5. Check the results in the file `users-updated.ldif` which is created in the same directory as the script, and verify that the appropriate changes were made. The changes contained in this file will automatically be made in the directory in the next step.

   Important Note: If there are users that you do not want to be managed by Identity Server, you should delete those entries from `users-updated.ldif` now. Then, instead of going on the to next step, manually load the file using the following command:

   ```
   ldapmodify -h hostname -p port -D bind_user -w password -a -c -f
     users-updated.ldif
   ```

6. In the script `update-users.pl`, uncomment the last line and replace variables appropriately. In the MadisonParc example, the last line of the script is changed from this:

   ```
   #system("$LDAP_MODIFY -h'$host' -p'$port' -D'$bind_user'
     -w'$bind_pwd' -a -c -f users-updated.ldif");
   ```

   to this:

   ```
   system("$LDAP_MODIFY -h'ginac.MadisonParc.com' -p'389'
     -D'cn=Directory Manager' -w'password' -a -c -f
     users-updated.ldif");
   ```

In Code Example 2-5, the user marker object class is indicated in bold.

Code Example 2-5 User Entry With User Marker Object Class

```
dn: uid=scarter, ou=People, dc=MadisonParc,dc=com
nsUniqueId: d8855082-1dd111b2-8024a6c9-802bec30
givenName: Sam
telephoneNumber: +1 408 555 4798
sn: Carter
ou: Accounting
ou: People
l: Sunnyvale
roomNumber: 4612
mail: scarter@MadisonParc.com
facsimileTelephoneNumber: +1 408 555 9751
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: inetuser
objectClass: inetadmin
objectClass: iplanet-am-managed-person
objectClass: iplanetPreferences
objectClass: iplanet-am-user-service
uid: scarter
```

Code Example 2-5 User Entry With User Marker Object Class *(Continued)*

```
cn: Sam Carter
userPassword: {SSHA}3XwjhBgbt6ae5syCndDeANoossEGRJ1NdnLyZw==
employeeType: Manager
departmentNumber: 1000
businessCategory: East
inetUserStatus: Active
```

Marking Static Groups

Static groups formed by adding uids to the group entry. To each group entry containing values for the `uniquemember` attribute, add the following object classes:

- `iplanet-am-managed-static-group`
- `iplanet-am-managed-group`

To Mark Static Groups Using the Sample Script

1. Copy `update-static-groups.pl` to the following directory:

   DirectoryServer_base`/shared/bin`

2. Set the `$base` variable to the base suffix of the directory tree to be managed by Identity Server. For example: `dc=MadisonParc,dc=com`.

3. In the directory where the script is located, at the command line enter the following:

   ```
   perl update-static-groups.pl
   ```

   When prompted, provide the following information:

   Enter Host Name: Enter the name of the computer system in which your Directory Server is installed.

   Enter Bind User Name: Enter a user name that has sufficient privileges for accessing the entire directory. For example: `cn=Directory Manager`

   Enter Bind password: Enter the password for the user you specified above.

   Enter port number: Enter the Directory Server port number. For example: `389`

4. Check the results in the file `static-groups-updated.ldif` which is created in the same directory as the script, and verify that the appropriate changes are listed. The changes contained in this file will automatically be made in the directory in the next step.

   Important Note: If there are static groups that you do not want to be managed by Identity Server, you should delete those entries from `static-groups-updated.ldif` now. Then, instead of going on the to next step, manually load the file using the following command:

   ```
   ldapmodify -h hostname -p port -D bind_user -w password -a -c -f
     static-groups-updated.ldif
   ```

5. In the script `update-static-groups.pl`, uncomment the last line, and replace variables appropriately. For example, in the MadisonParc example, the last line of the script is changed from this:

   ```
   #system("$LDAP_MODIFY -h'$host' -p'$port' -D'$bind_user'
     -w'$bind_pwd' -a -c -f static-groups-updated.ldif");
   ```

   to this:

   ```
   system("$LDAP_MODIFY -h'ginac.MadisonParc.com' -p'389'
     -D'cn=Directory Manager' -w'password' -a -c -f
     static-groups-updated.ldif");
   ```

In Code Example 2-6, marker object class for static groups is indicated in bold.

Code Example 2-6 Static Group Entry With Marker Object Classes

```
dn: cn=Directory Administrators, dc=MadisonParc,dc=com
nsUniqueId: 60a72e02-1dd211b2-8003a6c9-802bec30
objectClass: top
objectClass: groupofuniquenames
objectClass: iplanet-am-managed-group
objectClass: iplanet-am-managed-static-group
cn: Directory Administrators
uniqueMember: uid=kvaughan, ou=People, dc=MadisonParc,dc=com
uniqueMember: uid=alutz, ou=People, dc=MadisonParc,dc=com
uniqueMember: uid=gjensen, ou=People, dc=MadisonParc,dc=com
uniqueMember: uid=tcouzens, ou=People, dc=MadisonParc,dc=com
```

Marking Dynamic (Filtered) Groups

Dynamic or filtered groups are formed by building a search construct to find all user entries containing a specific attribute. These groups contain the `memberURL` attribute.

To each group containing the attribute `memberURL`, add the following object classes:

- `iplanet-am-managed-group`
- `iplanet-am-managed-filtered-group`

To Mark Filtered Groups Using the Sample Script

1. Copy `update-filtered-groups.pl` to the following directory:

   `DirectoryServer_base/shared/bin`

2. Set the `$base` variable to the base suffix of the directory tree to be managed by Identity Server.

   For example: `dc=MadisonParc,dc=com`

3. In the directory where the script is located, at the command line enter the following:

   ```
   perl update-filtered-groups.pl
   ```

4. When prompted, provide the following information:

   Enter Host Name: Enter the name of the computer system in which your Directory Server is installed.

   Enter Bind User Name: Enter a user name that has sufficient privileges for accessing the entire directory. For example: `cn=Directory Manager`

   Enter Bind password: Enter the password for the user you specified above.

   Enter port number: Enter the Directory Server port number. For example: `389`

5. Check the results in the file `filtered-groups-updated.ldif` which is created in the same directory as the script, and verify that the appropriate changes were made. The changes contained in this file will automatically be made in the directory in the next step.

   Important Note: If there are filtered or dynamic groups that you do not want to be managed by Identity Server, you should delete those entries from `filtered-groups-updated.ldif` now. Then, instead of going on the to next step, manually load the file using the following command:

   ```
   ldapmodify -h hostname -p port -D bind_user -w password -a -c -f
     filtered-groups-updated.ldif
   ```

6. In the script `update-filtered-groups.pl`, uncomment the last line in the `update-o.pl` file, and replace variables appropriately. In the MadisonParc example, the last line of the script is changed from this:

   ```
   #system("$LDAP_MODIFY -h'$host' -p'$port' -D'$bind_user'
     -w'$bind_pwd' -a -c -f filtered-groups-updated.ldif");
   ```

   to this:

   ```
   system("$LDAP_MODIFY -h'ginac.MadisonParc.com' -p'389'
     -D'cn=Directory Manager' -w'password' -a -c -f
     filtered-groups-updated.ldif");
   ```

In Code Example 2-7, marker object class for a filtered group is indicated in bold.

Code Example 2-7 Dynamic or Filtered Group With Marker Object Classes

```
dn: cn=North,ou=groups,dc=MadisonParc,dc=com
nsUniqueId: 60a72e35-1dd211b2-8003a6c9-802bec30
objectClass: top
objectClass: groupOfUniqueNames
objectClass: groupofurls
objectClass: iplanet-am-managed-group
objectClass: iplanet-am-managed-filtered-group
ou: groups
cn: North
memberURL:
ldap:///dc=MadisonParc,dc=com??sub?(&(|(objectclass=person)(objectc
 lass=groupofuniquenames))(businessCategory=*North*))
```

Marking Assignable Dynamic Groups

The *assignable* dynamic group is an Identity Server concept. In Identity Server, users in this type of group are typically allowed limited self-registration and account management privileges. In the MadisonParc example, users at the top level have administrators to create and manage their entries to comply with corporate specifications. Users under the Customers or Suppliers organizations are placed in assignable dynamic groups. The users can acquire membership by themselves when they log into the MadisonParc portal. Their membership entitles them to limited access to the MadisonParc portal; the information they provide at registration is minimal.

Add the following object classes to each dynamic group that you want to use as an assignable dynamic group in Identity Server:

- `iplanet-am-managed-group`
- `iplanet-am-managed-assignable-group`

To Mark Assignable Dynamic Groups Using the Sample Script

1. Copy `update-assignable-dynamic-groups.pl` to the following directory:

   DirectoryServer_base`/shared/bin`

2. Set the `$base` variable to the base suffix of the directory tree to be managed by Identity Server. For example: `dc=MadisonParc,dc=com`

3. In the directory where the script is located, at the command line enter the following:

   ```
   perl update-assignable-dynamic-groups.pl
   ```

4. When prompted, provide the following information:

   Enter Host Name: Enter the name of the computer system on which your Directory Server is installed.

   Enter Bind User Name: Enter a user name that has sufficient privileges for accessing the entire directory. For example: `cn=Directory Manager`

   Enter Bind password: Enter the password for the user you specified above.

   Enter port number: Enter the Directory Server port number. For example: `389`

5. Check the results in the file `assignable-dynamic-groups-updated.ldif` which is created in the same directory as the script, and verify that the appropriate changes were made. The changes contained in this file will automatically be made in the directory in the next step.

   Important Note: If there are assignable dynamic groups that you do not want to be managed by Identity Server, you should delete those entries from this `assignable-dynamic-groups-updated.ldif` now. Then, instead of going on the to next step, manually load the file using the following command:

   ```
   ldapmodify -h hostname -p port -D bind_user -w password -a -c -f
     assignable-dynamic-groups-updated.ldif
   ```

6. In the script `update-assignable-dynamic-groups.pl`, uncomment the last line in the `update-assignable-dynamic-groups.pl` file, and replace variables appropriately. In the MadisonParc example, the last line of the script is changed from this:

   ```
   #system("$LDAP_MODIFY -h'$host' -p'$port' -D'$bind_user'
     -w'$bind_pwd' -a -c -f
     assignable-dynamic-groups-updated.ldif");
   ```

   to this:

   ```
   system("$LDAP_MODIFY -h'ginac.MadisonParc.com' -p'389'
     -D'cn=Directory Manager' -w'password' -a -c -f
     assignable-dynamic-groups-updated.ldif");
   ```

Marking Group Containers

Group containers are organizational units (`ou`) that contain groups. To each group container that includes the `ou:Groups` attribute, add the following object class:

```
iplanet-am-managed-group-container
```

To Mark Group Containers Using the Sample Script

1. Copy `update-groups.pl` to the following directory:

   DirectoryServer_base`/shared/bin`

2. Be sure the `$base` variable is set to the base suffix of the directory tree to be managed by Identity Server.

   For example: `dc=MadisonParc,dc=com`.

   In the MadisonParc example, the script was also modified to include all group containers located under organizations. In Code Example 2-8, the script changes are indicated in bold.

Code Example 2-8 Scope in `update-groups.pl` is Modified.

```
# run search to find all group containers, putting their DNs in to a file
system("$LDAP_SEARCH -h \"$host\" -p \"$port\" -D \"$bind_user\"
   -w \"$bind_pwd\" -b \"$base\" -T \"(&(ou=groups)
     (!(objectclass=iplanet-am-*))(objectclass=organizationalunit))\
       " dn > group-container-updated.dn");
```

3. In the directory where the script is located, at the command line enter the following command:

   `perl update-groups.pl`

4. When prompted, provide the following information:

   Enter Host Name: Enter the name of the computer system in which your Directory Server is installed.

   Enter Bind User Name: Enter a user name that has sufficient privileges for accessing the entire directory. For example: `cn=Directory Manager`

   Enter Bind password: Enter the password for the user you specified above.

   Enter port number: Enter the Directory Server port number. For example: `389`

5. Check the results in the file `groups-updated.ldif` which is created in the same directory as the script, and verify that the appropriate changes were made. The changes contained in this file will automatically be made in the directory in the next step.

   Important Note: If there are group containers that you do not want to be managed by Identity Server, you should delete those entries from this `groups-updated.ldif` now. Then, instead of going on the to next step, manually load the file using the following command:

   `ldapmodify -h` *hostname* `-p` *port* `-D` *bind_user* `-w` *password* `-a -c -f groups-updated.ldif`

6. In the script `update-groups.pl`, uncomment the last line, and replace variables appropriately. In the MadisonParc example, the last line of the script is changed from this:

```
#system("$LDAP_MODIFY -h'$host' -p'$port' -D'$bind_user'
  -w'$bind_pwd' -a -c -f groups-updated.ldif");
```

to this:

```
system("$LDAP_MODIFY -h'ginac.MadisonParc.com' -p'389'
  -D'cn=Directory Manager' -w'password' -a -c -f
  groups-updated.ldif");
```

In Code Example 2-9, marker object class for a group under `dc=Customers` is indicated in bold.

Code Example 2-9 Group Container With Marker Object Class

```
...
dn: ou=Groups,dc=Customers,dc=MadisonParc,dc=com
nsUniqueId: 7880b101-1dd211b2-8007a6c9-802bec30
ou: Groups
objectClass: top
objectClass: organizationalunit
objectClass: iplanet-am-managed-group-container
...
```

Adding Custom Object Classes to Identity Server Schema

If your existing directory tree contains object classes you've created that do not come with Directory Server, then you'll have to add those object classes and attributes to the Identity Server schema. In the examples in this section, the MadisonParc directory tree uses two object classes and two user attributes that do not come with the Directory Server schema or with Identity Server schema. These object classes and attributes help to distinguish MadisonParc employees at the top level of the directory tree from non-employees in the Customers and Suppliers organizations. Before Identity Server can manage these extensions, changes must be made in the following three Identity Server files:

- `umExisting.xml`
- `amEntrySpecific.xml` (for organization data)
- `amUser.xml` (for user data)

This chapter contains detailed instructions for making these modifications. The instructions are provided here to help you see your existing data in Identity Server after you run the installer.

For background information about the Identity Server schema and detailed information about customizing Identity Server, see the *Java System Identity Server 2004Q2 Developer's Guide*.

Modifying the Creation Templates

The creation templates configure Identity Server to add or allow specific object classes and attributes when these entries are created. To expose custom object classes in the UI, you must modify the creation templates for both users and organizations in the `umsExisting.xml` file.

In the MadisonParc example, the existing directory tree has new object classes for both users and organizations.

The DAI Service

When you install Identity Server services, the `ums.xml file` is stored in Directory Server as the Directory Access Instructions (DAI) service. Identity Server will not allow you to load the `umsExisting.xml` file if the DAI service is already installed in Directory Server. Always remove the DAI service before modifying the `umsExisting.xml` file. Once you're finished modifying the files, you must reload the DAI service into Directory Server.

To Remove the DAI Service

Change to the `/bin` directory:

- Solaris systems: *IdentityServer_base* / bin
- Linus systems: *IdentityServer_base*/`SUNWam/bin`

Execute the following command:

```
./amadmin -u "user_naming_attibute=amadmin,ou=people,root_suffix"
  -w password -r   DAI
```

To Load the DAI Service

In the `/bin` directory, execute the following command:

```
./amadmin -u "user_naming_attibute=amadmin,ou=people,root_suffix"
  -w password -s umsExisting.xml
```

To Modify the Creation Templates

1. If during installation, you chose to load the Identity Server schema, or if you have already run the `amserveradmin` command for any reason, skip this step and go on to step 2. Otherwise, remove the DAI service.

   Change to the `/bin` directory:

   - Solaris systems: *IdentityServer_base* / bin
   - Linus systems: *IdentityServer_base*/`SUNWam/bin`

   Execute the following command:

   ```
   ./amadmin -u "user_naming_attibute=amadmin,ou=people,root_suffix"
     -w password -r DAI
   ```

2. Locate the following file. For example, on Solaris systems:

   `/etc/opt/SUNWam/config/umsExisting.xml`

3. Modify any custom naming attributes. For example, the MadisonParc directory tree uses the `domain` attribute instead of the `organization` attribute.

   Under the following SubConfiguration:

   `"BasicOrganization" id="CreationUmsObjects`

   change:

   `<Value>objectClass=organization</Value>`

   to:

   `<Value>objectClass=domain</Value>`

   In Code Example 2-10, bold indicates the changed value. Note that three lines down, the naming attribute `dc` was changed by Identity Server during installation.

Code Example 2-10 Changing the Organization Naming Attribute in the Creation Template

```
<SubConfiguration name="BasicOrganization" id="CreationUmsObjects">
                        <AttributeValuePair> <Attribute name="name" />
                            <Value>BasicOrganization</Value>
                        </AttributeValuePair>
                        <AttributeValuePair> <Attribute name="javaclass" />
                            <Value>com.iplanet.ums.Organization</Value>
                        </AttributeValuePair>
                        <AttributeValuePair> <Attribute name="required" />
                            <Value>objectClass=top</Value>
                                <Value>objectClass=domain</Value>
```

Code Example 2-10 Changing the Organization Naming Attribute in the Creation Template *(Continued)*

```
                        <Value>objectClass=sunManagedOrganization</Value>
                          <Value>objectClass=sunNameSpace</Value>
                          <Value>dc</Value>
                          <Value>inetdomainstatus=Active</Value>
                      </AttributeValuePair>
                      <AttributeValuePair> <Attribute
name="namingattribute"/>
                          <Value>dc</Value>
                      </AttributeValuePair>
                      <AttributeValuePair> <Attribute name="optional" />
                          <Value>*</Value>
                      </AttributeValuePair>
                  </SubConfiguration>
```

4. Add custom organization object classes.

In the MadisonParc example, madisonparc-org is added to the organization creation template. Under the following SubConfiguration:

`"BasicOrganiation" id="CreationUmsObjects">`

under the following element:

`<AttributeValuePair><Attribute name="required" />`

add the following:

```
<Value>objectClass=madisonparc-org</Value>
<Value>madisonparc-org-description</Value>
```

For example:

Code Example 2-11 Changing the Organization in the Creation Template

```
<SubConfiguration name="BasicOrganization" id="CreationUmsObjects">
                    <AttributeValuePair> <Attribute name="name" />
                        <Value>BasicOrganization</Value>
                    </AttributeValuePair>
                    <AttributeValuePair> <Attribute name="javaclass" />
                        <Value>com.iplanet.ums.Organization</Value>
                    </AttributeValuePair>
                    <AttributeValuePair> <Attribute name="required" />
                        <Value>objectClass=top</Value>
                        <Value>objectClass=domain</Value>
                        <Value>objectClass=sunManagedOrganization</Value>
                        <Value>objectClass=sunNameSpace</Value>
                        <Value>objectClass=madisonparc-org</Value>
                        <Value>dc</Value>
                        <Value>inetdomainstatus=Active</Value>
```

Code Example 2-11 Changing the Organization in the Creation Template

```
                    </AttributeValuePair>
                    <AttributeValuePair> <Attribute name="namingattribute"/>
                        <Value>dc</Value>
                    </AttributeValuePair>
                    <AttributeValuePair> <Attribute name="optional" />
                        <Value>*</Value>
                        <Value>madisonparc-org-description</Value>
                    </AttributeValuePair>
                </SubConfiguration>
```

5. Add custom user object classes.

   In the MadisonParc example, company is added to the user creation template. Under the following SubConfiguration:

   `"BasicUser" id="CreationUmsObjects">`

   under the following element:

   `<AttributeValuePair><Attribute name="required" />`

   add the following:

   `<Value>objectClass=company</Value>`

   For example:

Code Example 2-12 Adding Custom User Object Classes in the Creation Template

```
<SubConfiguration name="CreationTemplates" >
                  <SubConfiguration name="BasicUser" id="CreationUmsObjects">
                      <AttributeValuePair> <Attribute name="name" />
                          <Value>BasicUser</Value>
                      </AttributeValuePair>
                      <AttributeValuePair> <Attribute name="javaclass" />
                          <Value>com.iplanet.ums.User</Value>
                      </AttributeValuePair>
                      <AttributeValuePair> <Attribute name="required" />
                          <Value>objectClass=top</Value>
                          <Value>objectClass=person</Value>
                          <Value>objectClass=organizationalPerson</Value>
                          <Value>objectClass=inetOrgPerson</Value>
                          <Value>objectClass=iPlanetPreferences</Value>
                          <Value>objectClass=iplanet-am-user-service</Value>
                          <Value>objectClass=inetuser</Value>
                          <Value>objectClass=inetAdmin</Value>
                          <Value>objectClass=iplanet-am-managed-person</Value>
                          <Value>objectClass=company</Value>
                          <Value>cn=default</Value>
```

Code Example 2-12 Adding Custom User Object Classes in the Creation Template

```
                    <Value>sn=default</Value>
                    <Value>uid</Value>
                    <Value>inetuserstatus=Active</Value>
                </AttributeValuePair>
                <AttributeValuePair> <Attribute name="optional" />
                    <Value>*</Value>
                    <Value>companyname</Value>
                    <Value>acctname</Value>
                </AttributeValuePair>
                <AttributeValuePair> <Attribute name="namingattribute"/>
                    <Value>uid</Value>
                </AttributeValuePair>
            </SubConfiguration>
```

6. Reload the DAI service (the `ums.xml` file or `umsExisting.xml` file).

   In the `/bin` directory, execute the following command:

   ```
   ./amadmin -u "user_naming_attribute=amadmin,ou=people,root_suffix" -w password
     -s /etc/opt/SUNWam/config/ums/umsExisting.xml
   ```

Adding Attributes to the Organization Schema

To add attributes to the Organization schema, you must modify two services files:

- `amEntrySpecific.xml`
- `amEntrySpecific.properties`

The Identity Server Console uses the information in `amEntrySpecific.xml` for display purposes. Each Identity Server abstract entry may have a subschema in this XML file. In the following example, you would add the object class `external` to the organization subschema. If the directory tree contained customized organizational units, groups, or people containers, you would add or modify their subschemas in the same XML file.

The subschema name for an organizational unit will be `OrganizationalUnit`. The subschema name for a people container will be `PeopleContainer`.

NOTE The User subschema is not configured here in the `amEntrySpecific.xml` file, but in the `amUser.xml` file (see "Adding Attributes to the User Schema" on page 67.) Although any service XML file may describe an attribute that is only for a user, the `amEntrySpecific.xml` file can serve as a default place holder for user attributes that are not tied to a particular service.

The “any” attribute

The any attribute in the XML descriptions may have five possible values: `filter`, `display`, `adminDisplay`, `userReadOnly`, `required`, or `optional`. The values tell the Console whether the attribute should appear in the GUI. Typically, `required` and `optional` are not both displayed at the same time; they are mutually exclusive.

filter. The attribute is displayed in a search page.

display. The attribute is read / write for administrators and regular users.

adminDisplay. The attribute is read / write for administrators and is not displayed for regular users.

userReadOnly. The attribute is read / write for administrators but is read only for regular users. It is displayed as a label for regular users so that it is not editable. For example, the `display`, `adminDisplay`, and `userReadOnly` settings are used when displaying the user profile page and can be used to customize the page.

required. The attribute is displayed in the create page and requires a value during creation of the entry. If `any=required`, the attribute must have a value or the Console will not allow the Create operation. In the user interface, required fields are indicated with an asterisk (*). Use an empty string (" ") to tell the Administration Console to display nothing.

optional. The attribute is displayed in the create page but does not require a value during creation of the entry. If `any=optional`, the attribute will appear on the Create page without an asterisk. This would indicate that you don’t have to give it a value to create the entry. In the Create User page, the User ID is a required attribute but the First Name is optional.

In the following MadisonParc example, the attribute `madisonparc-org-description` will be displayed on the Organization page, and will be required for creation. This is indicated by the use of the `required` value. It will also be used on the Search page in Identity Server Console, as indicated by the use of the `filter` value.

```
<AttributeSchema name="madisonparc-org-description"
    type="single"
    syntax="string"
    any=required|filter
    i18nKey="o3"
/>
```

The “type” attribute

The *type* attribute can use a string, string list, single choice, multiple choice, or boolean value. For example, the `madisonparc-org-description` attribute can have only one of two descriptions: internal or external). You would make this attribute a single choice; each description would be one of the choices. The Identity Server Console would display a list containing only these cities. If multiple cities were allowed, the attribute could be a multiple choice.

To Add Attributes from a Custom Organization to the Organization Subschema

1. In the following file add the custom object class to the subschema Organization. For example, on Solaris systems:

   `/etc/opt/SUNWam/config/xml/amEntrySpecific.xml`

   In this example, the custom object class `madisonparc-org-description` was added to `amEntrySpecific.xml`.

```
<AttributeSchema name="madisonparc-org-description"
    type="single"
    syntax="string"
    any=required|filter
/>
```

2. In the same `amEntrySpecific.xml` file, create internationalization (i18n) keys (also called index keys or localization keys) for each attribute. All i18n Keys in an organization must be made up of unique strings. The Identity Server Administration Console will use this key to look up the display name for the attribute.

```
<AttributeSchema name="madisonparc-org-description"
    type="single"
    syntax="string"
    any="required|filter"
    i18nKey="o3"
/>
```

3. In the following file on Solaris systems:

   IdentityServer_base/SUNWam/locale/amEntrySpecific.properties

   Add the value for i18n Key you created in Step 2. This is the name that will be displayed in the graphical user interface. For example:

```
iplanet-am-entry-specific-service-description=Identity Server Entry Specific
g1=Member List
g2=Users Can Subscribe to this Group
dg1=Membership Filter
r1=Membership Filter
o1=Full DNS name
o2=Organization Status
o3=Organization Description
```

   All the attributes listed in the subschema are displayed in the Administration Console when an organization is displayed. If an attribute is not listed, the Administration Console will not display the attribute.

TIP If an attribute has no i18n Key, it will not be displayed on the administration console. If you add an attribute, and you don't see it in the administration console, be sure to check the i18n Key and properties.

4. Load all XML files.

   In the /bin directory, execute the following command:

   ./amserveradmin -u "*user_naming_attibute*=amadmin,ou=people,*root_suffix*" -w password

If you see any parsing errors, you should go back and double-check the changes you made in the previous steps. Also examine the syntax in the amEntrySpecific.xml file, and make sure you've used the correct syntax. If you need to look at syntax examples, look at the other service XML files located in the following directory on Solaris systems:

/etc/opt/SUNWam/config/xml

Adding Attributes to the User Schema

In this step, you will modify two files for services:

- amUser.xml
- amUser.properties

The amUser.xml file is where user attributes are described, just as organization and group schema are described in the amEntrySpecific.xml (see Step 2). The file amUser.xml describes the User service for Identity Server. Note that any service may describe an attribute that is for a user only. This file is just the default placeholder for user attributes that are not tied to a particular service.

When displaying a user's attributes, the Identity Server Administration Console gets all attributes from all services that are subschema type User, and displays them using the same values as used in the amEntrySpecific.xml file (see "The "any" attribute" on page 65 and "The "type" attribute" on page 66). In the following examples, a few attributes from the madisonparc-user object class are added to the file, thus it is not necessary to create a new service. It's only necessary to modify, or extend, the iplanetamuserservice.

Additional Notes About the amUser.xml File

The file amUser.xml contains a special attribute. The any=display attribute tells Identity Server whether to display the attribute in the user profile page. This is a misleading name since it implies access control. It is strictly used for display. If this attribute is set to no then the console will not display the attribute.

Also note that the attributes are defined under subschema User and not Dynamic. Any attribute defined under User is physically an attribute in the user entry. If you want the attribute to be a role-based or organization-based attribute, then you would define it under the Dynamic subschema. For detailed information, see the *Java System Identity Server 2004Q2 Developer's Guide*.

To Add Attributes from a Custom Organization to the User Subschema

1. In the following file, add the attributes from the custom object class to the User subschema on Solaris systems:

   `/etc/opt/SUNWam/config/xml/amUser.xml`

   For example, the following two attributes from the custom object class `company` were added to the file:

```
<AttributeSchema name="companyname"
    type=single
    syntax=string
    any=required|display
    />
<AttributeSchema name="acctnumber"
    type=single
    syntax=string
    any=required|filter|display
```

2. In the same `amUser.xml` file, create i18n Keys (also called *index keys* or *localization keys*) for each attribute. All i18n Keys in an organization must be made up of unique strings. The Identity Server Console will use this key to look up the display name for the attribute.

```
<AttributeSchema name="companyname"
    type=single
    syntax=string
    any=required|display
    i18nKey=u120
/>
<AttributeSchema name="acctnumber"
    type=single
    syntax=string
    any=required|filter|display
    i18nKey=u121
```

3. Add values for the i18n Keys you created in Step 2 to the following file on Solaris systems:

   IdentityServer_base/SUNWam/locale/amUser.properties

   For example:

```
iplanet-am-user-service-description=User
iwtUser-desc=Default User Profile
u101=UserId
u102=First Name
u103=Last Name
u104=Full Name
u105=Password
u106=Email Address
u107=Employee Number
u108=Telephone Number
u109=Manager
u110=Home Address
u111=User Status
u112=Account Expiration date (mm/dd/yyyy  hh:mm)
u113=User Authentication Configuration
u114=User Alias List
u115=Preferred Locale
u116=Success URL
u117=Failure URL
u118=Federation Information Key
u119=Federation Information
u120=Company Name
u121=Account Number
```

4. Load all XML files.

   In the /bin directory, execute the following command:

   ./amserveradmin -u "*user_naming_attibute*=amadmin,ou=people,*root_suffix*"
   -w password

If you see any parsing errors, you should go back and double-check the changes you made in the previous steps. Also examine the syntax in the amUser.xml file, and make sure you've used the correct syntax. If you need to look at syntax examples, look at the other service XML files located in the following directory on Solaris systems:

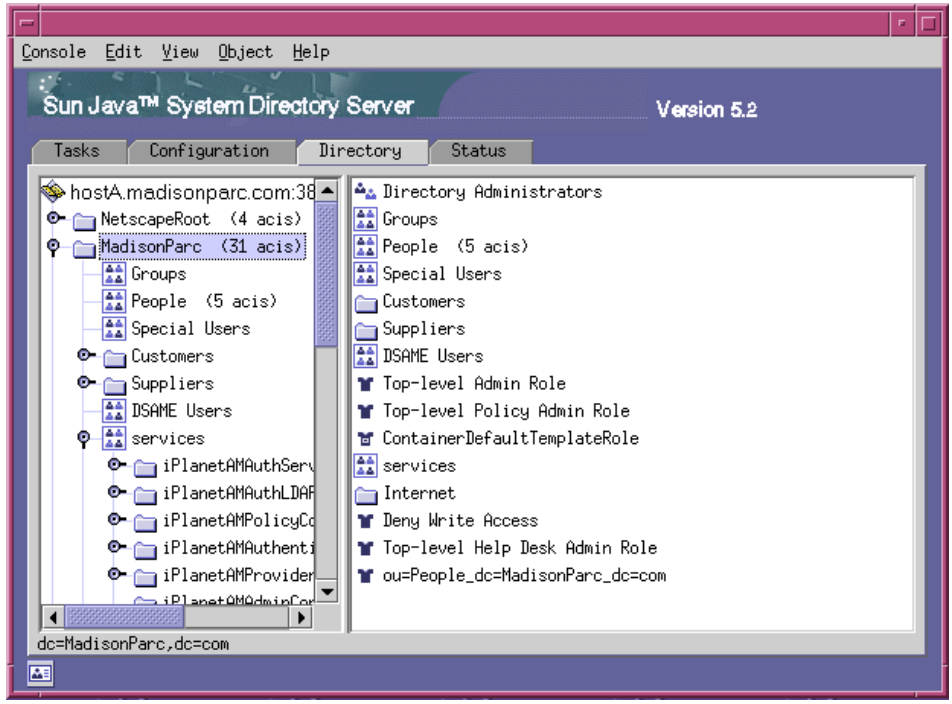
/etc/opt/SUNWam/config/xml

Loading Identity Server LDIF into Your Directory

Identity Server provides two different LDIF files to help you make the necessary modifications in your directory when you are enabling User Management services. You'll need to follow instructions for both loading `installExisting.ldif` and `install.ldif`.

Figure illustrates the MadisonParc directory tree after enabling both User Management and Policy Management services. Both `installExisting.ldif` and `install.ldif` files were loaded into an existing directory.

Figure 2-6 MadisonParc Directory Tree With Both `installExisting.ldif` and `install.ldif` Added



installExisting.ldif

The `installExisting.ldif` file contains Identity Server-specific entries that are loaded into Directory Server during installation. Typically, you will not need to modify this file before it gets loaded during the installation process.

You can use the `ldapmodify` utility that comes with Directory Server to load `installExisting.ldif`. In the MadisonParc example, when you load the LDIF, the following occurs:

- Users and marker object classes required for Identity Server are added to `dc=MadisonParc,dc=com` and to `dc=Customers` and `dc=Suppliers`.
- Default roles for organization and help desk administrators are created at the top level.
- Default Access Control Instructions (ACIs) for those administrator entries are set up.

To Load the installExisting.ldif File

1. Change to the `/ldif` directory. For example, on Solaris systems:

   ```
   cd /etc/opt/SUNWam/config/ldif
   ```

2. At the command line, enter the following:

   ```
   ldapmodify -v -c -D "cn=Directory manager" -w password -a
     -f installExisting.ldif
   ```

NOTE You must specify the `-c` option. Be sure you install only `installExisting.ldif`, and no other files in the same directory.

The Identity Server administration user `amAdmin` will be created under the `ou=People,dc=MadisonParc,dc=com` people container. This is the top level administrator for Identity Server. This administrator has read and write access to the entire `dc=MadisonParc,dc=com` root suffix. You can add one of your users to this top level administrator role after the Identity Server Console is started.

install.ldif

To Load the install.ldif File

1. Change to the `/ldif` directory. For example, on Solaris systems:

   ```
   cd /etc/opt/SUNWam/config/ldif
   ```

2. Enter the following command:

   ```
   ldapmodify -v -c -D "cn=Directory manager" -w password -a
   -f install.ldif
   ```

NOTE You must specify the `-c` option. Be sure you install only `install.ldif` and none of the other files in the same directory.

Results of Identity Server and Directory Modifications

After making the modifications in the previous steps, all entries in your existing directory will be manageable by Identity Server. The existing ACIs for the organization administrators do not have to be modified. Even though Identity Server uses roles and ACIs by default, your existing groups and ACIs will still work.

You can convert a groups-based directory tree to one that leverages roles and ACIs. If you choose to do this, you can use the Identity Server organization administrator roles and assign them to your existing `organizationList` administrators. For more information, see the *Java System Identity Server 2004Q2 Administration Guide.*

Appendix A

Identity Server 2004Q2 Upgrade Worksheets

This appendix contains the following worksheets to help you plan and perform an upgrade to Sun Java™ System Identity Server 2004Q2:

Pre-Upgrade Script Worksheet

For information about running the pre-upgrade script, see "Running the Pre-Upgrade Script" on page 15.

Table A-1 Pre-Upgrade Script (pre61to62upgrade) Worksheet

Script Option	Value and Example
Directory Server Host Name (fully qualified)	Your data: _____ Example: ds.example.com
Directory Server Port Number	Your data: _____ Example: 389 (default)
DN of the Identity Server Top-level Administrator (amadmin)	Your data: _____ Example: uid=amadmin,ou=people,dc=example,dc=com
Password for the DN of the Identity Server Top-level Administrator	Your data: _____
Backup Directory (where the script should back up Identity Server 6.1 files)	Your data: _____ Example: /opt/is_backup
Certificate Directory of the Web Container	Your data: _____ Example: /opt/SUNWwbsvr/alias

Identity Server 2004Q2 Installation Worksheets

Use the following worksheets before you run the Sun Java Enterprise System installer:

- Administration Worksheet
- Web Container Worksheets
- Directory Server Worksheets

For information about running the installer, see "Installing Identity Server 2004Q2" on page 16.

Administration Worksheet

Table A-2 Identity Server: Administration (1 of 6) Worksheet

Installation Option	Identity Server 6.1 (2003Q4) Value
Administrator User ID (amadmin)	Your data: _____ Example: amadmin (default)
Administrator User ID (amadmin) Password	Your data: _____
LDAP User ID (amldapuser)	_____ Example: amldapuser (default)
LDAP User ID (amldapuser) Password	Your data: _____
Password Encryption Key	Your data: _____ **Important** If you are deploying multiple instances of Identity Server, all instances must use the same password encryption key.

Web Container Worksheets

Table A-3 Identity Server: Web Container (2 of 6) Worksheet

Installation Option	Identity Server 6.1 (2003Q4) Value
Web Container	Your data: _____ Example: Sun Java System Web Server or Sun Java System Application Server

Table A-4 Identity Server: Sun Java System Web Server (3 of 6) Worksheet

Installation Option	Identity Server 6.1 (2003Q4) Value
Host Name	Your data: _____ Example: webhost.example.com
Web Server Port	Your data: _____ Example: 80 (default)
Web Server Instance Directory	Your data: _____ Example: /opt/SUNWwbsvr/https-myinstance
Document Root Directory	Your data: _____ Example: /opt/SUNWwbsvr/docs
Secure Server Instance Port	Your data: _____

Table A-5 Identity Server: Web Container for Running Identity Server Services (4 of 6) Worksheet

Installation Option	Identity Server 6.1 (2003Q4) Value
Host Name	Your data: _____
Services Deployment URI	Your data: _____ Example: amserver (default)
Common Domain Deployment URI	Your data: _____ Example: amcommon (default)
Cookie Domain	Your data: _____ Example: .example.com
Administration Console	Your response (required): Deploy new console
Console Deployment URI	Your data: _____ Example: amconsole (default)
Password Deployment URI	Your data: _____ Example: ampassword (default)
Console Host Name	Your data: _____ Example: console.example.com
Console Port	Your data: _____ Example: 80

Directory Server Worksheets

Table A-6 Identity Server: Directory Server Information (5 of 6) Worksheet

Installation Option	Identity Server 6.1 (2003Q4) Value
Directory Server Host (fully qualified)	Your data: _____ Example: ds.example.com
Directory Server Port	Your data: _____ Example: 389 (default)
Identity Server Directory Server Root Suffix	Your data: _____ Example: dc-example,dc=com
Directory Manager DN	Your data: _____ Example: ”cn=Directory Manager” (default)
Directory Manager DN Password	Your data: _____

Table A-7 Identity Server: Directory Server Information (6 of 6) Worksheet

Installation Option	Identity Server 6.1 (2003Q4) Value
Is Directory Server provisioned with user data?	Your response: Yes. Also, provide values for the marker and naming attributes in the next row.
Marker and Naming Attributes	Organization Marker Object Class Default: SunISManagedOrganization Your data: _____ Organization Naming Attribute Default: o Your data: _____ User Marker Object Class Default: inetorgperson Your data: _____ User Naming Attribute Default: uid Your data: _____

Post-Upgrade Script Worksheet

For information about running the pre-upgrade script, see “Running the Post-Upgrade Script” on page 18.

Table A-8 Post-Upgrade Script (Upgrade61DitTo62) Worksheet

Option	Value
Directory Server Hostname (fully qualified)	Your data: _____ Example: ds.example.com
Directory Server Port Number	Your data: _____ Example: 389 (default)
Directory Manager DN	Your data: _____ Example: ”cn=Directory Manager” (default)
Directory Manager DN Password	Your data: _____
DN of the Top-level Identity Server Administrator (amadmin)	Your data: _____ Example: uid=amadmin,ou=people,dc=example,dc=com
Password of the Top-level Identity Server Administrator	Your data: _____

Glossary

Refer to the *Sun Java™ Enterprise System Glossary* (http://docs.sun.com/doc/816-6873) for a list of terms that are used in this documentation set.

Index

SYMBOLS

A

B

C

D

F

G

I

J

L

M

O

P

R

S

T

U

W