

CA Identity Manager

Provisioning Guide

r12



Second Edition

This documentation and any related computer software help programs (hereinafter referred to as the "Documentation") is for the end user's informational purposes only and is subject to change or withdrawal by CA at any time.

This Documentation may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA. This Documentation is confidential and proprietary information of CA and protected by the copyright laws of the United States and international treaties.

Notwithstanding the foregoing, licensed users may print a reasonable number of copies of the Documentation for their own internal use, and may make one copy of the related software as reasonably required for back-up and disaster recovery purposes, provided that all CA copyright notices and legends are affixed to each reproduced copy. Only authorized employees, consultants, or agents of the user who are bound by the provisions of the license for the Product are permitted to have access to such copies.

The right to print copies of the Documentation and to make a copy of the related software is limited to the period during which the applicable license for the Product remains in full force and effect. Should the license terminate for any reason, it shall be the user's responsibility to certify in writing to CA that all copies and partial copies of the Documentation have been returned to CA or destroyed.

EXCEPT AS OTHERWISE STATED IN THE APPLICABLE LICENSE AGREEMENT, TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO THE END USER OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED OF SUCH LOSS OR DAMAGE.

The use of any product referenced in the Documentation is governed by the end user's applicable license agreement.

The manufacturer of this Documentation is CA.

Provided with "Restricted Rights." Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

Copyright © 2009 CA. All rights reserved.

CA Product References

This document references the following CA products:

- CA Identity Manager
- CA SiteMinder® Web Access Manager
- CA Security Command Center (SCC)
- CA Audit
- eTrust® Directory, also known as CA Directory

Contact CA

Contact Technical Support

For your convenience, CA provides one site where you can access the information you need for your Home Office, Small Business, and Enterprise CA products. At <http://ca.com/support>, you can access the following:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Provide Feedback

If you have comments or questions about CA product documentation, you can send a message to techpubs@ca.com.

If you would like to provide feedback about CA product documentation, complete our short [customer survey](#), which is also available on the CA support website, found at <http://ca.com/support>.

Contents

Chapter 1: Provisioning Overview 13

Multiple Accounts for Users	13
Identity Manager Users	13
Global Users	14
Accounts on Endpoint Systems	14
User Store and Provisioning Directory	15
Separate User Store and Provisioning Directories	15
Combined User Store and Provisioning Directory	16
Provisioning Server	17
Provisioning Manager	17
Administrator Authentication	18
Key Provisioning Terms	20
Endpoint Types (Namespaces)	20
Endpoints (Directories)	20
Account Templates (Provisioning Policies)	20
Connector Servers (Agent Plug-ins)	20
Connector Server Framework (Super Agent Framework)	21
Installation Directory	21

Chapter 2: Initial Configuration 23

Enable Provisioning	23
How to Configure an Environment for Provisioning	26
Import Custom Provisioning Roles	26
Configure the Inbound Administrator	26
Connect an Environment to the Provisioning Server	28
Configure Synchronization in the Provisioning Manager	28

Chapter 3: Managed Endpoints 31

How to Acquire Endpoints	31
Register an endpoint	31
Explore and Correlate an Endpoint	32
Log Endpoint Messages	33
Extending Custom Attributes on Endpoints	33
Create a New JIAM Jar File	33
Register the New JIAM Jar	36
Add the Attribute Element to Identity Manager	40

Chapter 4: Endpoint Accounts **41**

Account Management Tasks	41
View and Modify Endpoint Accounts	41
Assign Orphan Accounts	42
Assign System Accounts	43
Advanced Account Operations	43
Change the Global User for an Account	44
How Automatic Exploration Works	44
Delete Accounts	45
Use Delete Pending	45
Recreate Deleted Accounts	46

Chapter 5: Account Templates **47**

Account Templates Overview	47
Capability and Initial Attributes	48
Values and Rule Strings	49
Use Values Instead of Rule Strings	51
Default Account Template for an Endpoint Type	51
Advanced Rule Expressions	51
Rule Substrings	52
Combining Rule Strings and Values	52
Multivalued Rule Expressions	53
Explicit Global User Attribute Rules	55
Built-in Rule Functions	56
Program Exit Rule Functions	59
Account Template Management	59
Propagate Initial Attributes	60
How Capability Attributes are Synchronized	60
Attributes Only for New Accounts	64
Recompute Template Rules	64
etaultil and Account Templates	64
Change an account Template	65
Locate Non-Compliant Accounts	65
Synchronize Accounts with Account Templates	65

Chapter 6: Provisioning Roles **67**

Provisioning Roles Management	67
Create a Provisioning Role	67
Import a Provisioning Role	68
Order Provisioning Role Event Processing	69

Provisioning Role Performance	72
JIAM Object Cache	73
Session Pooling	74
IAMServer Initialization	74
Provisioning Roles for Existing Environments	74

Chapter 7: Passwords in Endpoint Accounts **75**

Choices for Password Management	75
Password Synchronization	75
Install the Password Synchronization Agent	76
How the Password Synchronization Agent Works	79
Account-Level Password Quality Checking	79
Password Quality Enforcement	81
Configure Password Synchronization	81
Failover	81
Enable Log Messages	81
Verify the Installation	82
Pluggable Authentication Module (PAM)	82
How PAM Works	83
Sample PAM Programs	83
Use PAM	84
Using PAM and Password Expiration	84
Update Global User Passwords	85
Use PAM with Multiple ADS Domains	86
Troubleshooting PAM	86
Configuration	88
UNIX Password Synchronization and PAM	93

Chapter 8: User Synchronization **99**

User Synchronization between Servers	99
Inbound Synchronization	99
Outbound Synchronization	99
Enable Password Synchronization	101
Endpoint Account to Global User Synchronization	102
Why Global Users Become Out of Sync	103
Global User Synchronization	104
Check User Synchronization	107
Synchronize Global Users or Roles	107
Synchronize Users in Create or Modify User Tasks	107
Account Synchronization	108
Check Account Synchronization	109

Synchronize Accounts	109
Chapter 9: etautil Batch Utility	111
Tasks You Can Perform	111
etautil Syntax	112
etautil Control Statements	114
Multivalued Attributes	119
Use DeletePending	120
Common Error Messages	121
Unknown error nnn opening Common Object Repository	121
End of file reached while expecting an operator	121
Object 'XXXX' operation failed: DB operation failed: Target DN not found.	121
Object 'XXXX' operation failed: No server plug-in found for operation	121
Class 'classname' is not a valid class name	122
Could not find keyword xxxxx for class classname	122
Obtain Operation Details	122
DOS Output from etautil	123
Chapter 10: SPML Service	125
SPML Overview	125
Benefits of Using SPML	125
When You Would Use the SPML Service	126
SPML Architecture	126
SPML Integration	130
Install SPML	132
Uninstall the SPML Service	132
SPML Service Configuration	132
Log On to the SPML Configuration Application	133
Add a New SPML Service	134
Modify an Existing Service	134
Rename an Existing Service	135
Delete an Existing Service	135
Configure SSL Support for Tomcat Servers	136
Configure SPML Client Computer to Support SSL Security	138
CMDRA Commands	139
SPML Feed	143
Using the SPML Manager's Templating Functionality	148
Using Velocity Templates	150
Retrying SPML Requests	155

Chapter 11: Sample SPML Requests **161**

Request Execution Types	161
Request Types	162
Add Request	162
Batch Request	162
Cancel Request	164
Delete Request	164
Extended Request	165
Modify Request	169
Propagate Global User Changes	170
Schema Request	171
Search Request	172
Status Request	174
Global Settings	174
Example: Search for Attributes Defined in Global Settings	175
Example: Modify Attributes in Global Settings	176
Account Containers	176
Example: Create an Account Container	177
Example: Create an Account within a Sub-Container	177
Complex Attributes	178
Example: Add a Single-Valued Complex Attribute	179
Example: Add a Multivalued Complex Attribute	179
Request Retries	180
Propagate Global User Changes	180
Example: Modify a Global User and Propagate Changes to Associated Accounts	181
Example: Modify Complex Attribute and Propagate Changes to Accounts	181
Escaping Special Characters in Object Identifiers	182
Escaping Special Characters in Search Filters	182

Appendix A: Provisioning Server Maintenance **183**

Back Up and Restore the eTrust Directory Database	183
Fine-tune the eTrust Directory Database	184
Performance Tuning with notSearchable Attributes	185
View and Maintain Log Files	185
Server Event Logging	186
Diagnostic Logging	187
Shut Down the Provisioning Server service	192

Appendix B: Provisioning Servers on UNIX **193**

No UNIX GUI Clients or Utilities	193
--	-----

Command Line Examples	194
Libraries and Executables	194
Registry Access	195
Parser Tables	196
UNIX Services for Provisioning	196
Working with Hung or Crashed Servers	197
Scheduling Periodic Actions	197
Passwords on Command Lines	198
Server Event Logging Destinations	198
Program Exit Definitions	198

Appendix C: Advanced Configuration Options **201**

Advanced Configuration Options Overview	201
Global Properties	203
Domain Configuration	204
Provisioning Directory Parameters	206
Authentication Parameters	207
Authorization Parameters	207
Cache Parameters	208
Compatibility Parameters	214
Configuration Setup Parameters	215
Connections Parameters	215
Endpoint Parameters	218
Explore and Correlate Parameters	222
Identity Manager Server Parameters	230
Operation Details Parameters	233
Password Synchronization Parameters	236
Password Parameters	237
Processes Parameters	240
Processor Parameters	242
Search Parameters	243
Servers Parameters	248
Statistics Parameters	248
Synchronization Parameters	250
Transaction Log Parameters	254

Appendix D: Program Exits **259**

Program Exits Overview	259
Ordering of Program Exit Invocations	260
Basic Structure of Program Exits	262
Define Common Exits in the Provisioning Manager	262

Chapter 1: Provisioning Overview

For installations of Identity Manager with a Provisioning Server, additional functionality is available to provide users with additional accounts.

This section contains the following topics:

[Multiple Accounts for Users](#) (see page 13)

[User Store and Provisioning Directory](#) (see page 15)

[Provisioning Server](#) (see page 17)

[Key Provisioning Terms](#) (see page 20)

[Installation Directory](#) (see page 21)

Multiple Accounts for Users

For any user in Identity Manager, you can associate that user with other accounts if you have installed a Provisioning Server. For example, a user may need an Exchange account for email, an Ingres account for database access, and an Active Directory account to use a Windows system. When you assign a provisioning role to a user, that user receives the accounts defined by the account templates in the provisioning role.

The account templates define specific characteristics of the account. For example, a template for an Exchange account could define the size of the mailbox.

Note: To install the Provisioning Server, see the *Installation Guide*.

Identity Manager Users

Identity Manager users are the users who are visible through an admin task, such as View User or Modify User. Some users may have an associated global user, a corresponding account which exists on the Provisioning Server.

Global Users

A global user is an object maintained by the Provisioning Server. It corresponds to one person or other identity that needs access to the Provisioning Server or the endpoints that it manages. A global user object contains information such as the person's name, password settings, job title, phone number, and address. The primary purpose of a global user is to tie together a person's accounts on the endpoint systems.

Accounts on Endpoint Systems

Each global user may have accounts on endpoint systems, such as Microsoft Exchange, Ingres, or Active Directory. If the global user is also associated with an Identity Manager user, the Identity Manager user has access to the accounts on the endpoint systems.

When you view or modify a user in the User Console, the Accounts tab lists the endpoint accounts for that user. For each account, the tab displays information such as the account name, the endpoint where the account exists, and the status of the account. For a modify task, additional options are available for actions such as changing a users password and locking or suspending an account.

View User: *ken.davis*

Profile **Admin Roles** **Provisioning Roles** **Accounts**

Search for an Account of Endpoint Type Windows NT

where + Name = * - + Search Clear

☐ Select	▲ Name	Endpoint Type	Endpoint	EndPoint Description	Container	Suspended
☐	ken.davis	Windows NT	aj-joy-vm.ca.com	Windows NT Endpoint.	aj-joy-vm.ca.com	Active
☐	ken.davis	Windows NT	iam-fw-wl9	test	iam-fw-wl9	Active

For details on the other options you can provide on the Accounts tab, see the user console help for the Configure Accounts tab.

Two other types of accounts exist on endpoint systems:

- Orphan Accounts—Accounts that are not associated with a global user
- System Accounts—Accounts that are not associated with a global user and are used to manage the endpoint system

These accounts can be assigned to a user through the Manage Orphan Accounts and Manage System Accounts tasks in the User Console.

User Store and Provisioning Directory

To provide options for managing users and automatic provisioning of additional accounts for those users, Identity Manager coordinates two user stores:

- The *Identity Manager user store*, the user store maintained by Identity Manager. Typically, this is an existing store that contains the user identities that a company needs to manage.

The user store can be an LDAP directory or a relational database.

In the Management Console, you create an Identity Manager Directory object to connect to the user store and to describe the user store objects that Identity Manager will maintain.

- The *Provisioning Directory*, the user store maintained by the Provisioning Server.

It is an instance of eTrust Directory and includes global users, which associate users in the Provisioning Directory with accounts on endpoints such as Microsoft Exchange, Active Directory, and Ingres.

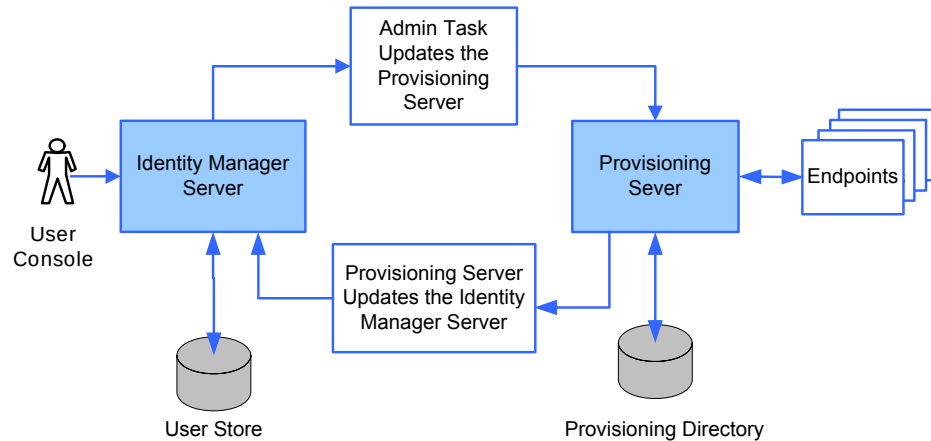
Only some Identity Manager users have a corresponding global user. When a Identity Manager user receives a provisioning role, the Provisioning Server creates a global user.

Separate User Store and Provisioning Directories

The following figure shows a separate user store and Provisioning Directory. In this figure:

- An Identity Manager administrator uses an admin task that edits a user in the user store, which affects the Provisioning Directory.
This change may also update an endpoint (such as an email server) which has a connector to the Provisioning Server.
- A change made in the Provisioning Server (or an endpoint with a connector to the Provisioning Server) updates the Identity Manager user store and Provisioning Directory.

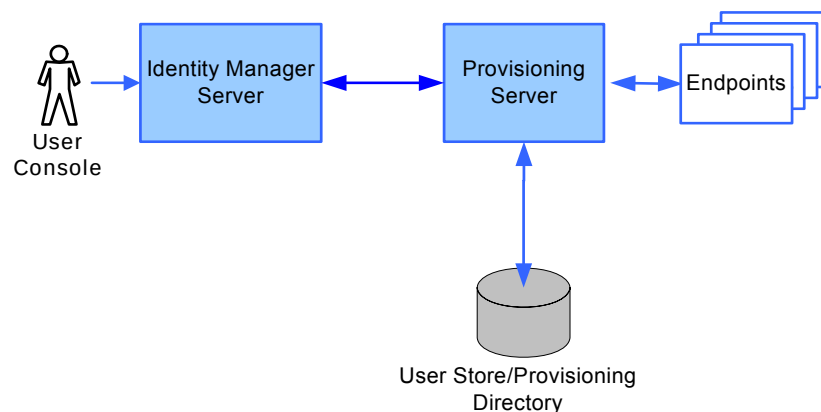
For example, an endpoint, such as a Human Resources application, might update the email addresses of users.



Combined User Store and Provisioning Directory

If you select an instance of eTrust Directory for both user stores, the directory functions as both a user store and Provisioning Directory. Users you create in Identity Manager are stored in that directory, but you can still modify that directory as a Provisioning Directory. For example, you can select the user attribute to use in the Provisioning Server for a specific user attribute used by Identity Manager.

The following figure shows the use of one directory for both the user store and Provisioning Directory.



In this situation, changes to the user store/Provisioning Directory can be initiated by Identity Manager, the Provisioning Server, or an endpoint.

Note: When Identity Manager uses a combined user store and Provisioning Directory, organizations are not supported. If you need to manage organizations, use a separate user store and Provisioning Directory. How to Choose a User Store Solution lists additional differences between combined and separate user store/Provisioning Directory implementations.

Provisioning Server

The Provisioning Server is the server that manages additional accounts that are assigned to an Identity Manager user. When you assign a provisioning role to an Identity Manager user, the Provisioning Server creates accounts on endpoints that meet the requirements of the role. For example, if you assign a provisioning role that includes an Exchange account template, the Provisioning Server assigns an Exchange account to the user.

Provisioning Manager

The Provisioning Manager is the user interface for managing endpoint types, such as Exchange or Oracle, and endpoints, such as a specific system where Exchange is installed. This interface was formerly called eTrust Admin Manager.

Other capabilities exist in the Provisioning Manager, such as exploring and correlating accounts; however, that additional functionality is now duplicated in the Identity Manager User Console, where it is more easily accessed. The Provisioning Manager method will not be available in future releases, so we recommend using the Identity Manager User Console instead.

To start using Provisioning Manager, choose Start, CA, Identity Manager, Provisioning Manager.

Note: The first time you use Provisioning Manager, you need to perform the procedure in the chapter about initial configuration.

Administrator Authentication

Objects in the provisioning domain are protected at several different levels, but overall access to the domain is protected by authentication security, which requires all administrators to identify themselves. The global user name and password that the administrator enters are checked against information stored in the provisioning directory.

You can configure the Provisioning Server to request authentication with a native system. For more information, see the Pluggable Authentication Module (PAM) in the chapter "Passwords in Endpoint Accounts."

Administrator Login

The first time you log on to the Provisioning Manager, you use the `etaadmin` global user, whose password was set up when the Provisioning Server was installed.

The Provisioning Server also provides other built-in global users that provide authentication information for use only by Provisioning Server components. You will not use these global users to log on; instead, they provide additional authentication information for the domain.

The `etaadmin` user is similar to a built-in superuser account. You set a password for this user when the Provisioning Server is installed. It is *imperative* to remember that password because you need it to log on as `etaadmin` the first time you use Provisioning Manager.

When you log on using the `etaadmin` object, you have access to all the objects in the domain. You should immediately create a global user object for yourself and assign the `DomainAdministrator` profile to it. When this user has been created, you should log in as that user and not perform any more actions as the non-specific `etaadmin` administrator. Avoiding the use of `etaadmin` improves the traceability of actions as seen in various logs. For more security, you can delete or suspend the `etaadmin` user after you create your own account.

Administrator Authorization

Authorization determines what administrators can do on the Provisioning Server. It defines the privileges that an administrator has in a domain. You can authorize an administrator by assigning an *admin profile* to the administrator's global user object or by assigning an admin profile to one of the administrator global user groups.

Admin Profiles

Admin profiles permit administrators certain types of access and privileges to manage objects in a domain. Admin profiles contain all the privileges that administrators need to perform different tasks. While administrative privileges can be assigned directly to global users, using admin profiles provides several advantages:

- Several administrators can be defined to a profile, and therefore, each receives the same administrative privileges.
- The operations that an administrator is allowed to perform will typically necessitate being granted a long list of administrative privileges. Placing them into an admin profile is less error prone as it lets you define the profile once and then apply those privileges to multiple administrators.
- Admin profiles can be accessed from other domains, making it easy for other administrators to create new profiles from existing profiles in other domains.

Note: When assigning individual administrative privileges, you must give the administrator Read access to the object and its container. This access is necessary to list and search for objects.

Default Admin Profiles

The Provisioning Server provides default admin profiles that control the privileges of an administrator. These profiles give administrators access to the objects in the domain of the profile. Like the default administrator objects, such as etaadmin, the following profiles are created automatically when you install the Provisioning Server:

- **DomainAdministrator**-Gives administrators full access to every object in the domain. Administrators who have this profile in the root domain have full access to all Provisioning Server objects and security information.
- **PasswordAdministrator**-Lets administrators change passwords and activate or suspend global users.
- **UserAdministrator**-Lets administrators manage users in the domain. Administrators with this profile cannot modify provisioning roles or account templates.
- **ReadAdministrator**-Lets administrators read every object in the domain.
- **SelfAdministrator**-Defines the actions that can be performed by self-administrators. By default, this profile authorizes self-administrators to read their own global user object, list their accounts, and modify specific attributes of their own global user or accounts. You can customize this profile to meet your self-administrator authorization requirements.

Note: With the exception of SelfAdministrator, you cannot modify or delete these profiles.

Key Provisioning Terms

In Provisioning Manager and the Identity Manager User Console, you see some new more consistent terms that integrate the names in these user interfaces.

Endpoint Types (Namespaces)

An endpoint type is a specific type of endpoint system, such as Microsoft Exchange or Oracle, that is managed by Identity Manager. Prior to r12, an endpoint type was called a namespace.

Endpoints (Directories)

An endpoint is a specific installation of an endpoint type, such as Microsoft Exchange installed on the server, Eastern. Prior to r12, an endpoint was called a Directory.

Account Templates (Provisioning Policies)

You can create account templates in the Provisioning Manager. These templates provide the basis for accounts on a specific endpoint type. Prior to r12, an account template was called a Provisioning Policy.

Connector Servers (Agent Plug-ins)

The Provisioning Server communicates with endpoint systems through connector servers and connectors. You can install connector servers on the same system as the Provisioning Server or a remote system.

Two types of connector servers exist:

C++ Connector Server (formerly a SuperAgent)

A connector server that can run C++ connectors (formerly called agent plug-ins).

Java Connector Server

A connector server that can run Java connectors.

If you were to look at what is running on a system (task manager on Windows, ps on Unix) you would see just the connector servers, not the individual connectors.

The connector servers perform all the common tasks, such as listening for incoming requests and communicating using the LDAP protocol. The connector communicates with the endpoint system, processing the data that is sent and received.

Note: While the C++ Connector Server is described in this guide, the Java Connector Server provides additional functionality that requires separate Java Connector Server Guides. All guides are part of the CA Identity Manager bookshelf.

Connector Server Framework (Super Agent Framework)

The connector server framework enables the Provisioning Server for the domain to assign specific connector servers to communicate with specific endpoint systems. Prior to r12, the connector server framework was called the Distributed Super Agent Framework. In r12, the command-line tool that manages the Connector Server Framework has been renamed from dsfconfig to csfconfig.

Installation Directory

The path notation *PSHOME* is used throughout this guide to refer to the Provisioning Server installation directory. The following is the default directory on Windows:

`\Program Files\CA\Identity Manager\Provisioning Server`

Note: *PSHOME* is a notation of convenience only, and does not represent a defined system environment variable.

Chapter 2: Initial Configuration

The initial configuration you perform for provisioning involves enabling provisioning and configuration for each Identity Manager environment.

This section contains the following topics:

[Enable Provisioning](#) (see page 23)

[How to Configure an Environment for Provisioning](#) (see page 26)

Enable Provisioning

Note: Before you can enable provisioning, you install the provisioning directory on eTrust Directory. For more information, see the *Installation Guide*.

To enable Provisioning

1. Open the Management Console by typing the following URL in a browser:

`http://hostname:port/idmmanage`

hostname

Defines the fully qualified host name of the system where the Identity Manager server is installed

port

Defines the application server port number.

2. Click Directories.

The Identity Manager directories window appears. You use this window to configure the provisioning directory to work with Identity Manager.

3. Click New to start the directory wizard.

4. Type the path and filename of the directory configuration XML file for configuring the provisioning directory, or browse for the file. Click Next.

The directory configuration XML file for the provisioning directory is installed in the following location:

im_admin_tools_dir\directoryTemplates\ProvisioningServer

im_admin_tools_dir is the location of the Identity Manager Administrative Tools, which is:

For Windows—[set Installation Path variable]

For UNIX—[set the alternate Installation Path variable]

Note: You can use this directory configuration file as installed with no modification.

5. Supply values for the fields on this window as follows:

Name

Is a name for the provisioning directory that you are configuring.

Description

(Optional) Describes the Identity Manager directory.

Connection Object Name

Specifies a name for the user directory.

- If Identity Manager does not integrate with SiteMinder, specify a meaningful name for the object used by Identity Manager to connect to the user directory.
- If Identity Manager integrates with SiteMinder, you have two choices:

If you want to create a user directory connection object in SiteMinder, specify any meaningful name. Identity Manager creates this object in SiteMinder with the name you specify.

If you want to connect to an existing SiteMinder user directory, specify the name of the SiteMinder user directory connection object exactly as it appears in the Policy Server user interface.

Host

Specifies the host name or IP address of the system where the user directory is installed.

Port

Specifies the port number of the user directory.

Provisioning Server Domain

Specifies the name of the provisioning domain that Identity Manager will manage.

The name must match the name of the provisioning domain that you specified during installation.

Note: The domain name is case-sensitive.

Username

Specifies a global user that can log into the Provisioning Manager.

The user must have the Domain Administrator profile, or an equivalent set of privileges for the Provisioning Domain.

Password

Specifies the password for the global user that you specified in the Username field.

Confirm Password

Enter the password that you typed in the Password field again for confirmation.

Secure Connection

Indicates whether Identity Manager uses a secure connection.

Be sure to select this option for Active Directory user stores.

Note: The fields that appear on this window depend on the user store type and the information you provided in the directory configuration file in Step 4. If you provided values for any of these fields in the directory configuration file, Identity Manager does not prompt you to supply these values again.

Click Next.

6. Review the settings for the Provisioning Directory. Click Previous to make changes.

Status information is displayed in the Directory Configuration Output window.

7. Click Finish to start environment configuration, which is described in [How to Configure Inbound Synchronization](#) (see page 26).

How to Configure an Environment for Provisioning

To configure an Identity Manager environment for provisioning, you select Environments, New in the Management console and answer the questions. The creation of the environment is described in the chapter on managing Identity Manager environments in the *Configuration Guide*.

Then, you create a special Identity Manager user, called the Inbound Administrator, create a connection to the Provisioning Server, and synchronize Provisioning Manager with the environment.

Note: Whenever you make changes to provisioning properties for an environment, you need to restart the application server for the changes to take effect.

Import Custom Provisioning Roles

When you create the environment, you have the choice to use the default roles or a custom role definition file you create. If you import custom roles definitions, you must *also* import the Provisioning Only role definitions. After creating the environment, import the role definitions from this folder:

IdentityMinder.ear\management_console.war\WEB-INF\Template\environment

Configure the Inbound Administrator

For inbound synchronization to work, you create a special Identity Manager user called the *inbound administrator*. At previous releases of CA Identity Manager, the inbound administrator was called the *corporate user*. No user logs into this user account; instead, it is used internally by Identity Manager. However, you need to create this user account and give it the appropriate tasks.

Inbound Event	Task
☐ POST_ADD_GLOBAL_USER	Provisioning Create User
☐ POST_MODIFY_GLOBAL_USER	Provisioning Modify User
☐ POST_ENABLE_GLOBAL_USER	Provisioning Enable/Disable User
☐ POST_DISABLE_GLOBAL_USER	Provisioning Enable/Disable User
☐ POST_CHANGE_GLOBAL_USER_PWD	Provisioning Modify User

To configure the inbound administrator

1. Log into the Identity Manager environment as the user with the System Manager role.
2. Create a user. You might name the user **inbound** as a reminder of its purpose.
3. Choose Admin Roles, Modify Admin Roles and select a role that contains the tasks you use for synchronization.
 - Provisioning Create User
 - Provisioning Enable/Disable User
 - Provisioning Modify User

Note: If you have made no changes to the default synchronization tasks, use the Provisioning Synchronization Manager role.

4. On the Members tab, add a member policy that includes the following:
 - A member rule that the new user meets.
 - A scope rule that provides access to all users who are affected by provisioning directory changes that trigger inbound synchronization.



Owners can modify the role.

Owner Rules

Owner Rule		
	where (User ID = "inbound")	

5. In the Management Console:
 - a. Select the Environment.
 - b. Select Advanced Settings, Provisioning.
 - c. Complete the Organization for Creating Inbound Users field if the Identity Manager directory includes an organization.

This organization is where users are created when inbound synchronization occurs. For example, when a user is added to the provisioning directory, Identity Manager adds the user to this organization.

- d. Complete the Inbound Administrator field with the User ID of the user that you created in Step 2.
- e. Click Validate to confirm the user ID is accepted as shown in the following example where the complete user ID appears below the user ID entered.

Organization for Creating Inbound Users	ou=NeteAuto,dc=securit Validate
	Unique Name: ou=NeteAuto,dc=security,dc=com
Inbound Administrator	uid=SuperAdmin,ou=Pec Validate
	Unique Name: uid=SuperAdmin,ou=People,ou=Employee,ou=NeteAuto,dc=security

- f. Make changes to other fields on this screen. No changes are required.

If you do make changes, be sure that you understand how the fields interact. For details on each field, click the Help link on the screen.

Connect an Environment to the Provisioning Server

To connect an environment to the Provisioning Server

1. In the Management Console, click Environments.
A list of existing environments appears.
2. Click the name of the environment that you want to associate with the Provisioning Server.
3. Click the right arrow icon in the Provisioning Server field.
The Provisioning Properties screen opens.
4. Select the Provisioning Server.
5. Click Save at the bottom of the screen.
6. [Configure Synchronization in the Provisioning Manager](#) (see page 28).

Configure Synchronization in the Provisioning Manager

Inbound synchronization keeps Identity Manager up to date with changes that occur in the provisioning directory. Changes in the provisioning directory include those made using Provisioning Manager and changes in endpoints for which the Provisioning Server has a connector.

You can configure different inbound synchronization for each Environment that requires provisioning support. For example, the Customers and Employees Environments may require inbound synchronization for different events. In the Customers environment, synchronization takes place when a global user is added or modified in the Provisioning Server. Only password changes are synchronized in the Employees Environment.

To configure synchronization for an Environment

1. Choose Start, CA, Identity Manager, Provisioning Manager.
2. Click System, Identity Manager Setup.
3. Complete the Host Name field with the name of the system where the Identity Manager Server is installed.
4. Complete the Port field with the application server port number.
5. Complete the Environment name field with the alias for the environment.
6. Click Add.
7. Repeat steps 3-6 for each Environment that is associated with the Provisioning Server.
8. If this is the first Environment, fill in the Shared Secret fields using the password entered during Identity Manager installation for the user for embedded components.

Note: These fields do not apply if FIPS is enabled in this installation.

9. Set the Log Level as follows:
 - No Log--No information is written to the log file.
 - Error--Only error messages are logged.
 - Info--Error and information messages are logged (default).
 - Warning--Error, warning, and information messages are logged.
 - Debug--All information is logged.
10. Restart the application server before you log in to the environment.

Note: For a log of inbound synchronization operations and any problems encountered during synchronization, see the following file:

`PSHOME\\logs\etanotify<date>.log`

Chapter 3: Managed Endpoints

This section contains the following topics:

[How to Acquire Endpoints](#) (see page 31)

[Log Endpoint Messages](#) (see page 33)

[Extending Custom Attributes on Endpoints](#) (see page 33)

How to Acquire Endpoints

Acquiring an endpoint automatically populates the provisioning directory with accounts and other objects found in the acquired endpoint. An *acquired* endpoint is any application or computer managed by Identity Manager. Acquiring an endpoint consists of registering it, exploring it, and correlating it. Acquiring an endpoint may also include creating global users automatically and setting global user attributes from account attributes.

The following is the procedure to acquire an endpoint:

- Register it in the provisioning directory. When you register an endpoint, you are declaring it as an endpoint managed by Identity Manager.
- Explore its contents. When you explore an endpoint, Identity Manager finds the objects in the endpoint and stores instances of them in the provisioning directory.
- Correlate the explored accounts. When you correlate accounts on an endpoint, Identity Manager associates them with a global user in the provisioning directory. You may choose whether the correlate function creates any global users that are not present or whether it associates accounts with no matching global user to the [default user] global user.
- Update global user fields from accounts. Whether or not you chose to let correlate function create missing global users, you can optionally run an additional function that sets (or refreshes) selected global user attributes using account's attribute values that you select.

Register an endpoint

When you *register* an endpoint, you declare a new endpoint for the server to manage. The registration process uses the endpoint property sheet for the endpoint type. For example, if you have a Windows 2003 server named WINPC7, you must register the WINPC7 computer as an endpoint under the Active Directory endpoint type on the server to manage this computer. See Register the Primary Directory and Register the Secondary Directory in the Provisioning Manager help.

Explore and Correlate an Endpoint

In the User Console, you can define an explore and correlate definition. Then, you use this definition with the Execute Explore and Correlate task.

To create an explore and correlate definition

1. In an Identity Manager environment, choose Endpoints, Explore and Correlate Definitions, Create Explore and Correlate Definition.
2. Click Okay to start a new definition.
3. Complete the Explore and Correlate Tab as follows:
 - a. Select Container/Endpoint to choose a container (if this endpoint has containers) and an endpoint to explore.

The explore and correlate process includes any container you select and its sub-containers. For a directory container, it includes all the containers in the sub-tree.
 - b. Fill in Explore and Correlate name with any meaningful name.
 - c. On the first tab, choose the operations to perform:
 - **Explore directory for Manage Objects**—Find objects that are stored on the endpoint and not in the provisioning directory
 - **Correlate Accounts to Global Users**—Correlate the objects that were found in the explore process to the users in the provisioning directory. If the user is found, the object is correlated with the user. Otherwise, you can choose if the user is created or attached to the default user.
 - **Update global user fields**—If a mapping exists between the object fields and the user fields, the user fields are updated with data from the objects fields.
4. Complete the Recurrence tab
 - a. Click Schedule.
 - b. Complete the fields that appear to determine when the explore and correlate operation should occur.

You may prefer to schedule this operation to execute overnight to interfere less with routine access of the system.
5. Click Submit.

To use an explore and correlate definition

1. In an Identity Manager environment, choose Provisioning Endpoints, Execute Explore and Correlate.
2. Choose an explore and correlate definition to execute.
3. Click Submit.

The user accounts that exist on the endpoint are created or updated in Identity Manager based on the explore and correlate definition you created.

Log Endpoint Messages

In Identity Manager, each endpoint has a logging component associated with it. Its function is to log tracing messages that allow you, or support technicians, to monitor requests being sent to the managed endpoint by the Provisioning Server.

You can configure each endpoint's logging component to send messages to a text file, called Provisioning Server. You can also specify that it receives only messages of a certain severity level, such as error, warning, or fatal messages.

The text file can also be used to can log Provisioning Manager messages (set through File/Preferences in the Provisioning Manager) or Provisioning Server messages (set in Global Properties in the Provisioning Manager System Task). However, the Provisioning Server messages represent the official transaction log of the Provisioning Server. Use these messages to record a permanent audit trail of requests received and processed by the Provisioning Server.

Extending Custom Attributes on Endpoints

The Provisioning Server can manage custom endpoint attributes. To enable Identity Manager to read custom endpoint attributes that are associated with provisioning roles, there are additional procedures required. This section describes the concepts and the required procedures for custom endpoint attributes.

Create a New JIAM Jar File

After you add the custom attributes to your connector, create a new JIAM Option within Identity Manager

Note: For more information on adding custom attributes, see the Connector Guide.

The JIAM Jar File

Java Identity and Access Management (JIAM) is a Java front end to the Provisioning Server.

You can extend the IAM system through the connectors by loading the connector's JAR files.

These JAR files can be built in the connector GUI directory by calling NMAKE, located in the makefile in the PSDKHOME/samples/sdk/gui/ subdirectory.

The following is an example that builds the JAR file for the SDK Sample connector:

```
JIAMJAR=1
NAMESPACE=SDK
...
ptts :: $(ADMIN)/Data/sdkparse.ptt
...
! IF "$(JIAMJAR)" == "1"
  jars :: $(ADMIN)/jiam/jiamExt$(NAMESPACE).jar
! ENDIF
```

JIAMJAR

Variable that enables or disables the build for the JIAM JAR file. The value 1 means true, and any value other than 1 means false.

NAMESPACE

Name of the connector (SDK, in this case).

You can modify the GUI makefile for your connector to include content similar to that in the example to build the JAR file.

Note: The second block must be put after the "ptts" target because the parser table is required to build the JIAM JAR file.

Generating JAVA Source Files

RDTutility.bat is a tool called by NMAKE to generate the JIAM JAR file. Although the process of generating the JAR file is hidden, you can call the RDTutility.bat directly without using NMAKE. In this way, you have more control over it. For example, you might choose to retain the JAVA source files and the XML files for the connector.

You can specify a config file containing extra information in the PSDKHOME/bin/config to change the default behaviors of the RDT.

To specify a config file to change default behaviors of the RDT

1. Generate Java Objects other than Endpoint, Account, and Policy objects. For example, adding the following lines to the LDAconfig.xml instructs the RDT to generate the LDAP Group Java object as well.

```
<namespace name=LDA">
<objectClass name="LdapGroup"/>
</namespace>
```

2. Change the name of attributes used by JIAM to be different from the one defined in the parser table. The following example shows that groupNames are used as the bean property name for the attribute eTLDAMemberOf for LDA connector.

```
<namespace name=LDA">
  <objectClass name="LdapAccount">
    <property>
      <name>groupNames</name>
      <attr>eTLDAMemberOf</attr>
    </property>
  </objectClass>
</namespace>
```

The following example assumes that you have developed a new connector called PKI and that you want to generate the JAVA source files for it.

To generate JAVA source files with the RDTutility.bat batch file

1. Copy the compiled parser table for the connector (should be PKIparse.ptt in this case) into the following directory:

PSDKHOME/data

2. Open the DOS command prompt and change to the directory that you want to use as the base directory to contain the source and XML files.
3. Set the JAVASDK environment variable to point to the JAVA SDK directory by entering the following command:

```
set JAVASDK=C:\j2sdk1.4.2_04
```

Note: This command sets the variable temporarily. If you want to set the variable permanently, make it a system environment variable.

4. Provide extra mapping file (PKIconfig.xml) in the PSDKHOME/bin/config directory.

5. Enter the following command:

```
"PSDKHOME/bin/RDTutility.bat" PKI TRUE
```

PKI

Acronym for the new option.

TRUE

Signifies that the output files are kept.

6. The RDTutility creates a directory called output in the current directory, and places all the JAVA source files and XML files in the following directory structure:

```
/output/PKI/IAM/ObjLayer/
```

Note: The jiamExtPKI.jar file is also generated and copied to the PSDKHOME/jiam directory.

RDUtility.bat Syntax

The following is the syntax for the RDTutility.bat:

```
RDUtility.bat namespaceName keepOutput
```

namespaceName

Acronym for the connector, for example, ADS or N16.

keepOutput

Determines whether to keep the output files. Specify either TRUE or FALSE. The default value is FALSE, if no value is specified.

Extended LDAP JIAM Jar

Extending the Provisioning Server LDA schema requires regenerating the JIAM LDA classes and replacing them in the JIAM.jar. These class files for the LDA connector are in a separate jar called jiamExtLDA.jar, which needs to be added to the classpath of any applications using JIAM. The jiam.jar file includes it in the classpath of its manifest file, so applications have access to it automatically.

Register the New JIAM Jar

Once you have created the new JIAM Jar, register the new JIAM Jar within Identity Manager.

How to Define JIAM Custom Connector Extensions

Each custom connector contains an `ExtensionDescriptor` object that defines the metadata necessary for use with the JIAM API. This object is packaged in the connector's jar file. To configure JIAM to support a custom object, you specify the `ExtensionDescriptor` object's class name in a `JIAMExtensions` property when JIAM is initialized by Identity Manager.

In the Management Console, define the `JIAMExtensions` property in the Miscellaneous Properties (in Advanced Settings) for an Identity Manager environment. Specify the property as follows:

Property Name: `JIAMExtensions`
Description: Defines available custom connector extensions
Value: Fully-qualified `ExtensionDescriptor` class name.

Note: To add support for multiple custom connectors, specify the `ExtensionDescriptor` class name for each connector in a space-delimited list.

For example, the Provisioning SDK includes a sample connector called "SDK." When you build this sample, the optional build step creates a jar file named `jiamExtSDK.jar`. The fully qualified name of the `ExtensionDescriptor` object within that jar is:

```
com.ca.iam.model.options.sdk.impl.SDKOptionDescriptor
```

To use this connector in an Identity Manager environment, create the `JIAMExtensions` property as follows:

Property: `JIAMExtensions`
Value: `com.ca.iam.model.options.sdk.impl.SDKOptionDescriptor`

Note: Do not add the `.class` suffix to the value.

Identity Manager deployments that require custom connector objects must define this property for each of the custom connectors they want to manage in the Identity Manager environment. This applies to Identity Manager environments that directly manage a Provisioning Server and a Provisioning Directory, and to environments that include a Identity Manager user store and a provisioning directory.

How to Deploy JIAM Custom Connector Jar Files

For Identity Manager to manage custom connectors, each `ExtensionDescriptor` object defined in the `JIAMExtensions` list, and the associated custom connector jar must be defined in the classpath. Since Identity Manager runs in an application server, each JIAM extension jar must be present on an application server-specific classpath.

Deployment on WebLogic or WebSphere

For WebLogic and WebSphere application servers, you have to extract the connector jar file before the application server can use it.

To deploy a custom connector jar file on WebLogic or WebSphere

1. Extract the contents of each connector jar file to the following Identity Manager EAR folder:
 - For WebLogic:
`weblogic_home/weblogic_domain/Identity Minder.ear/custom`
 - For WebSphere:
`websphere_home/installedApps/server_Instance/Identity Minder.ear/custom`
2. Restart the application server.

Deployment on JBoss

JBoss application servers automatically load any jar files that are placed in the `jboss_home\server\default\lib` directory. You can use this feature to load a custom connector jar file.

To deploy custom connector jars on Jboss

1. Copy each connector jar file to the `jboss_home\server\default\lib` directory.
2. Restart the application server.

How to Validate Custom Connectors

To validate that each custom connector is initialized properly, define an additional miscellaneous property in the Management Console to perform a validation step. This property and its values are shown below:

Property Name:	JIAMExtensionsValidate
Description:	Validates that Identity Manager can load each of the JIAM Extension Descriptor objects defined in the JIAMExtensions property

Values:

- True: Attempts to instantiate each of the defined descriptor objects. If the object can not be loaded, it is not added to the list of JIAM Extensions. The custom connector objects can not be used.
- False: No validation occurs. Instantiation of the descriptor object is delayed until a reference to any custom connection object is made. Failures at run-time may prevent use of certain tasks until this condition is resolved.

Note: For performance reasons, you should set the value of this property to 'False' after validating the JIAM extensions.

To troubleshoot any custom connector load failures, set the following Debug category in the IdentityMinder.ear\config\com\netegrity\config\log4j _<app server>.properties file:

```
log4j.category.ims.llsdk.etrustadmindirectory=DEBUG
```

```
log4j.category.ims.llsdk.etrustadmindirectory=true
```

Add the Attribute Element to Identity Manager

Once the custom attribute is defined in JIAM, edit the endpoint type IM2JIAM mapping file, which is located in the following directory:

IdentityMinder.ear\custom\provisioning\im2jiammapping

Add the new attribute element within the mapping file and restart the application server. For example,

```
<attribute imname="custom01" etname="custom01" type="String"
ismultiValued="false" permission="READWRITE" displayname="Patient Code"
description="Patient Code" />
```

imname

Name of the attribute that will be referenced in the Identity Manager Server

etname

Name of the attribute referenced in JIAM

type

Type of attribute. Valid values are String, Integer, and Collection

isMultiValued

Defines if the attribute is a multi-valued attribute.

permission

Permission of the attribute. Valid values are READWRITE, READONLY, and WRITEONCE

displayname

Display name of the attribute

description

Description of the attribute

Chapter 4: Endpoint Accounts

In the User Console, you can modify accounts associated with users and manage orphan and system accounts, which are accounts not currently associated with Identity Manager.

In the Provisioning Manager, you can perform a number of additional operations on accounts.

This section contains the following topics:

[Account Management Tasks](#) (see page 41)

[Advanced Account Operations](#) (see page 43)

Account Management Tasks

In the User Console, you can view and modify accounts and assign system and orphan accounts to an Identity Manager user.

View and Modify Endpoint Accounts

Tasks that allow you to view a user's profile, such as View User or Modify My Profile, include an Accounts tab that lists that user's accounts on endpoints.

View User: ken.davis

ProfileAdmin RolesProvisioning RolesAccounts

Search for an Account of Endpoint Type Windows NT

where + Name = * - + Search Clear

☐ Select	▲ Name	Endpoint Type	Endpoint	EndPoint Description	Container	Suspended
☐	ken.davis	Windows NT	aj-joy-vm.ca.com	Windows NT Endpoint.	aj-joy-vm.ca.com	Active
☐	ken.davis	Windows NT	iam-fw-wl9	test	iam-fw-wl9	Active

For each account, Identity Manager displays information such as the account name, the endpoint where the account exists, and the status of the account. For a modify task, additional options are available for changing a users password and locking or suspending an account.

In this example, the Accounts tab includes a Search button, which means the tab is configured with a search screen. You can configure this tab to use a list screen, a search screen, or both.

- When both the screens are configured, the search screen determines the fields in the search results.
- If only a list screen is configured, it determines the fields in the search results.
- If neither screen is configured, the accounts tab uses a static list display, which means that the Accounts tab cannot be customized for display columns.

For details on the other options you can provide on the Account tab, see the User Console help for the Configure Accounts tab.

Assign Orphan Accounts

In the User Console, you can manage orphan accounts, which are accounts not associated with a global user.

To create a default user for orphan accounts

If you have a separate Provisioning Directory, you need to create the Provisioning Server default user in the Identity Manager user store. The default user is used for orphan accounts.

1. In the User Console, click the Users tab.
2. Choose Manager Users, Create User.
3. Name the user as follows, including the brackets:
[default user]

You can now assign orphan accounts to users.

To assign an orphan account

1. In the User Console, choose Endpoints,
2. Choose Manage Orphan Accounts.
3. Search for and select a user.
4. Choose an Identity Manager user to assign to the orphan user.

Assign System Accounts

In the User Console, you can manage system accounts, which are accounts that are not associated with a global user and are used to manage the endpoint system.

To assign a system account to a user, you create an admin task based on the Manage System Accounts task. This task needs to be completed with a user or duplicated and a user added to the copy of the task.

The new task has a specific Identity Manager user who applies for a specific endpoint. You could create one task for each type of endpoint.

To configure a task to assign system accounts

1. In the User Console, choose Roles and Tasks, Admin Tasks, Create Admin Task.
2. Base the new task on the Manage System Accounts
For example, you could create a task called *Manage Oracle System Accounts* to assign system accounts on an Oracle endpoint type.
3. On the Tabs tab, select Configure Manual Correlation and enter the User ID of the user who should be assigned to this system account.
4. Submit the task.
5. Include this task in a role.
6. Assign the role to a user who should assign system accounts for an endpoint to a user.

The user with this role can execute the new task to assign system users to an Identity Manager user.

Advanced Account Operations

In the Provisioning Manager, you can perform a number of additional operations on accounts:

- Associate an Account with Different Global Users
- Automatically Explore Accounts
- Delete Accounts
- Use Delete Pending
- Recreated Deleted Accounts

Change the Global User for an Account

The following are instances when you would want to associate an account to a different global user:

- You have two global users with the same name and Identity Manager correlates the account to the wrong person
- Identity Manager correlated an account to the [default user] object and you want to associate it with another global user object
- You created an account using New and now you want to associate it with a global user

To associate an account with a different global user in the Provisioning Manager, drag and drop the account onto the correct global user.

How Automatic Exploration Works

The addition or deletion of accounts or other objects using tools native to the endpoint are unnoticed by Identity Manager until the next time you explore the endpoint. The exploration process notices additions and deletions (and in some cases modifications) that have occurred and applies those changes to the Identity Manager representation of the object in the provisioning directory.

However, if you use the Provisioning Manager to attempt to create an object with the same name before this exploration occurs, Identity Manager notices an object with that name already exists and report this error. Identity Manager then explores that object, creating a representation of it in the provisioning directory. You can immediately start working with that object. The automatic one-object explore occurs whenever an Add, Move or Rename operation generates an already exists error from the endpoint when the object does not exist in the provisioning directory.

This automatic exploration feature can be combined with the feature controlled by the Synchronize/Automatic Correlation domain configuration parameter. For more information, see the chapter "Advanced Configuration Options." When these two features work together, they first process an attempt to create an account from an account template as an attempt to create a new account. Then the processing uses the following steps:

- Notices an unexplored account
- Explores that account automatically
- Correlates the account automatically to the global user
- Adds an account template to the account as though it were an existing account correlated to this global user.

Delete Accounts

If you must delete an account, you can use the following methods in the Provisioning Manager:

- Right-click the account and select Delete
- Right-click a global user and select Delete User and Accounts
- Run the Delete Accounts wizard
- Synchronize global users with provisioning roles and specify that you want to delete extra accounts

When you remove a global user from a provisioning role, the Provisioning Manager provides these choices for account deletion:

- If you decide to delete these accounts, Identity Manager removes the accounts from the provisioning directory.
- If you decide not to delete the accounts, you can use the Synchronize User with Roles option and select the Delete Account.

When you remove a global user from a provisioning role before deleting accounts, you can list the accounts for the global user. Right-click the global user and select List Accounts.

- The account listing displays the provisioning roles to which each account belongs. If an account belongs to one provisioning role, it is deleted when you remove that user from that role and accept the user synchronization action to delete the accounts.
- If an account belongs to no provisioning role, it is an extra account and is reported by Check User Synchronization. The account is deleted if you select the Synchronize the User with Roles menu item on the global user.

Use Delete Pending

Identity Manager can be configured, on an endpoint-by-endpoint basis, so accounts on an endpoint are not deleted when administrators initiate delete or sync actions that would normally delete the accounts. Instead, the accounts are placed in a Delete Pending state in Identity Manager and in a Suspended state on the managed endpoint.

Delete Pending accounts can be identified in the Provisioning Manager on the Statistics tab of the account properties. A suspended account has a suspend reason of Delete Pending and a timestamp when it entered this state. The storing of the Delete Pending status and Suspended timestamp permits the writing of a utility that identifies these Delete Pending accounts and deletes them from the Provisioning Server and the managed endpoint later.

Recreate Deleted Accounts

If you delete an account on a managed endpoint by using a tool other than Identity Manager, the Check Account Synchronization feature reports the account as missing, because it exists in the Provisioning Directory but not on the managed endpoint. When this happens, recreate the account on the endpoint by issuing the Synchronize Account with Account Templates function, which recreates the account using the account templates associated to the account.

If accounts are recreated, Identity Manager logs them as recreated. These accounts can be identified separately from accounts that have been updated because administrators need to be aware that attributes other than capability attributes (for example, passwords) have been set to the original account template values.

Chapter 5: Account Templates

You can create account templates in the Provisioning Manager. These templates provide the basis for accounts on a specific endpoint type. Account templates provide the same capabilities as Provisioning Policies did in previous releases of the Admin Manager.

This section contains the following topics:

[Account Templates Overview](#) (see page 47)

[Default Account Template for an Endpoint Type](#) (see page 51)

[Advanced Rule Expressions](#) (see page 51)

[Account Template Management](#) (see page 59)

[etaultil and Account Templates](#) (see page 64)

Account Templates Overview

To simplify account management, you create and maintain accounts using account templates, which can be associated with one or more provisioning roles. Account templates contain the attributes that are used to create accounts. You can define attributes using rule strings or values.

Using account templates, you can:

- Control what account attributes global users have on an endpoint when their accounts are created
- Combine account attributes from different provisioning roles, so global users have only one account, on a specific endpoint, with all the necessary account attributes
- Create or update account attributes as global users change provisioning roles
- Synchronize account attributes so global users have only the attributes they need
- Perform queries to see which accounts are to be created, updated, or deleted during a synchronization operation
- Determine which account attributes can be synchronized with provisioning roles and which cannot

Capability and Initial Attributes

Account templates include two types of attributes:

- *Capability attributes* represent account information, such as storage size, quantity, frequency limits, or group memberships. Provisioning Manager bolds the capability attributes on all account template screens to make identifying capability attributes easy.
- *Initial attributes* represent all information that is initially set for an account, such as account name, password, and account status and personal information such as name, address and phone numbers.

Accounts are considered synchronized with their account templates when all the capability attributes are synchronized. These are attributes that differ from endpoint-type to endpoint-type such as group memberships, privileges, quotas, login-restrictions, which that control what the user can do when logging into the account.

Other account attributes are not updated by synchronization. They are initialized from the account templates during account creation. And they can also be updated during propagation functions. The Provisioning Server provides two propagation functions (an immediate update of accounts at the time the account template is changed and an update of accounts at the time global user attributes change).

Values and Rule Strings

Account attributes are generated using rule strings, which are variables substituted with the value for the specific account. Rule strings are useful when you want to generate attributes that may change from one account to another. When rules are evaluated, Identity Manager replaces the rule strings entered in the account templates with data specified in the global user object.

Note: Rule evaluation is not performed on accounts created during an exploration or on accounts created without provisioning roles.

Account names must be unique, so you should enter a %AC% rule string in the account name field on an account template. When Identity Manager creates an account using this account template, it will use the global user's account name.

The following table lists the rule strings in Identity Manager:

Rule String	Description
%AC%	Account name
%D%	Current date in the format <i>dd/mm/yyyy</i> (the date is a computed value that does not involve the global user's information). This rule string is equivalent to one of the following: %\$\$DATE() % %\$\$DATE%
%EXCHAB%	Mailbox hide from exchange address book
%EXCHS%	Mailbox home server name
%EXCMS%	Mailbox store name
%GENUID%	Numeric UNIX/POSIX user identifier. This rule variable is the same as %UID% as long as the global user's UID value is set. However, if the global user has no assigned UID value, and UID-generation is enabled (Global Properties on System Task), the next available UID value is allocated, assigned to the global user, and used as the value of this rule variable.
%P%	Password
%U%	Global user name
%UA%	Full address (generated from street, city, state, and postal code)

Rule String	Description
%UB%	Building
%UC%	City
%UCOMP%	Company name
%UCOUNTRY%	Country
%UCUxx% or %UCUxxx%	Custom field (xx or xxx represents the 2- or 3-digit field ID as specified on the Custom User Fields tab in the System Task frame)
%UD%	Description
%UDEPT%	Department
%UE%	Email address
%UEP%	Primary email address
%UES%	Secondary email addresses
%UF%	First name
%UFAX%	Facsimile number
%UHP%	Home page
%UI%	Initials
%UID%	Numeric UNIX/POSIX User Identifier
%UL%	Last name
%ULOC%	Location
%UMI%	Middle initial
%UMN%	Middle name
%UMP%	Mobile telephone number
%UN%	Full name
%UO%	Office name
%UP%	Telephone number
%UPAGE%	Pager number
%UPC%	Postal code, zip code
%UPE%	Telephone number extension
%US%	State
%USA%	Street address
%UT%	Job title

Use Values Instead of Rule Strings

To use a specific, constant value for an account attribute, enter the value in the account template field instead of in a rule string. For example, you can enter values for specifying frequency limits or quantity size.

If the constant attribute value must contain more than one percent sign, enter two percent signs (%%) each time. Identity Manager translates them to one percent sign (%) when building the account attribute value. If the account template value contains only one percent sign, Identity Manager does not generate an error. The rule states that if you want a literal value of 25%, you must specify 25%%. However, as a special case, 25% will be accepted.

Default Account Template for an Endpoint Type

When you install a connector, the installation process often creates one or more default account templates for that endpoint type. A default template defines the minimum security level needed to access an endpoint of that endpoint type. You can assign this template to a provisioning role or use it as a model to create other account templates for the endpoint type. Some endpoint types have multiple default templates to create different types of accounts.

Note: For more information about default templates for endpoint types, see the Connector Guide that describes the endpoint type.

Advanced Rule Expressions

To provide greater flexibility than simple global user attribute substitution, you can enter advanced rule expressions, including the following:

- Substrings of rule expressions using Offset and Length
- Combinations of rule strings and values
- Rule expressions to set multiple values for multivalued account attributes
- Rule variables for other global user attributes
- Invocation of Built-in functions
- Invocation of customer-written Program Exit functions

Rule Substrings

The following is the syntax for creating a substring value of a rule variable:

`%var[:offset,length]%`

var

Represents the name of the predefined rule variable as defined in the table shown previously.

offset

(Optional) Defines the starting offset of the substring suffix. The number 1 represents the first character.

length

(Optional) Defines the ending offset of the substring suffix. A length value of asterisk (*) indicates to the end of the value.

For example, to set an account attribute to the first 4 characters of a global user's Building attribute, use the following to define the variable:

`%UB:1,4%`

If the Building attribute is empty or has fewer than four characters, the resulting account attribute value will have fewer than four characters.

Combining Rule Strings and Values

You can combine rule strings and constant values into an account template attribute value. For example, if there were no %UI% rule string, you could obtain the same effect by concatenating multiple rule expressions as follows:

`%UF:,1%UMI:,1%UL:,1%`

The %UA% rule string is equivalent to the following:

`%USA%, %UC%, %US%, %UPC%`

You can also combine a rule string with a constant value to create a UNIX home endpoint attribute as follows:

`/u/home/%AC%`

Multivalued Rule Expressions

Most rule expressions are single-valued. They start from one global user attribute value (possibly empty) and result in one account attribute value (also possibly empty). However, sometimes you want to consider an empty global user attribute as 0 values. Sometimes you may want to generate multiple values to populate a multivalued account attribute value.

The following rule syntax lets you work with zero or more values that a global user attribute may contain:

`%*var%`

The optional multivalued flag asterisk (*) immediately after the first percent sign % of a rule expression indicates that the result of this rule expression should be 0, 1 or more than 1 value depending on how many values the referenced global user attribute contains.

Most global user attribute values are single-valued, so they may only contain 0 or 1 values. However, the custom attributes (CustomField01 through CustomField99) are multivalued attributes, so a rule variable referencing these attributes may contain 0, 1, or more than 1 value.

If a global user attribute has more than 1 value, but you fail to include the asterisk (*) in your rule expression, then the result of the rule evaluation will be that of the first value. However, in most cases attribute values are officially unordered and as a result the value that Identity Manager considers first may not be predictable.

If a global user attribute has more than one value, and you include the * in your rule expression, multiple values are generated for the account attribute. Do not define such a multivalued rule expression in an account template if the account attribute being set from that account template attribute is not itself multivalued.

You can define an extended account attribute in the ADS endpoint type to be multivalued; and use this multivalued rule expression syntax to set that attribute. For example, consider an environment that defines an extended ADS account attribute named patents and custom global user attribute number three also named patents.

An ADS account template could define, for the patents attribute, the rule string `%*UCU03%`. Then, you could change a global user's patents attribute by adding one or more values. When applying the changes to the global user, select the option of updating the global user's accounts. This consults the account's account template, finds the rule variable `%*UCU03%`, and knows to copy all of the global user's patents to the account's patents attribute.

Similarly, during account creation, rule strings are evaluated. Furthermore, during account template change, if the rule string has been changed, you can choose to recompute the rule for all accounts associated to the account template.

The `%*var%` syntax is also meaningful for variables *var* that refer to single-valued global user attributes. This is true only when concatenation is involved and if the referenced attributes are unset on global users.

The optional multivalued flag asterisk (*) indicates that the rule containing a `%*var%` rule variable evaluates to no value if the global user attribute has no values. This is different from the single-valued rule expression `%var%`, which always evaluates to a single value, even if it is an empty string.

To understand this difference, consider the following rule strings:

```
(310)%UP%  
(310)%*UP%
```

Both rule strings appear to append area code 310 to the telephone number. However, they are different because if global users have no value for their telephone number, the first rule evaluates to the account value of (310). The second rule string generates no value and leaves the account attribute unset.

On the other hand, consider the following rule strings that appear to append the telephone extension to the telephone number:

```
%UP% %UPE%  
%UP% %*UPE%
```

If everyone has a telephone number, but some do not have extensions, the first rule string generates a value that includes the phone number for each global user with no extension. The second rule string generates no values. In this case, use the first rule with `%UPE%`.

Explicit Global User Attribute Rules

Each global user has many more attributes than are listed in the previous rule table. You will probably have no need to create rule expressions referencing any of these other attributes. However, should the need arise, you can use the following syntax to refer to a specific global user attribute:

`%#ldap-attribute%`

For instance, if you must determine the value of the global user's Suspended field, you would determine the corresponding LDAP attribute name for this field (which is `eTSuspended`) and create the rule expression that evaluates to 0 or 1, like `eTSuspended`:

`%#eTSuspended%`

As another example, you can obtain the global user's assigned provisioning roles with the following rule expression:

`%*#eTRoleDN%`

These provisioning roles are full LDAP distinguished name values. Perhaps in conjunction with the built-in function `RDNVALUE` (see the table that follows), the values would be a little more useful. Note the multi-value indicator asterisk (*) so as to obtain all of the global user's assigned provisioning roles as multiple values.

The substring syntax is also applicable to these rule expressions, so you could use `%#eTTelephone:6,*%` to mean the same thing as `%UP:6,*`. Each asks Identity Manager to strip off the first five characters of the global user's telephone field.

Built-in Rule Functions

You may use built-in rule functions in your rule expressions to perform various transformations on the values. The general form of built-in rule function invocation is

```
%[*]$$function(arg[,...])[:offset,length]%
```

where the multivalued indicator asterisk (*) and the offset and length substring specifications are once again optional.

The recognized built-in functions are as follows:

Built-in Rule Function	Description
ALLOF	Merges all the parameters into a multivalued attribute. Order is preserved and duplicates are removed. For example, if global user attributes are set to the following: <pre>eTCustomField01: { A, B } eTCustomField02: { A, C }</pre> Then, the rule: <pre>%*ALLOF(%*UCU01%,%*UCU02%)%</pre> evaluates to three values { A, B, C }.
DATE	Evaluates to the current date in <i>dd/mm/yyyy</i> format. The rule expression %D% is equivalent to one of the following: <pre>\$\$\$DATE() % \$\$\$DATE%</pre>
FIRSTOF	Returns the first value of any of the parameters. Used to insert a default value if an attribute is not set: <pre>\$\$\$FIRSTOF(%UCU01%,'unknown')%</pre> <pre>\$\$\$FIRSTOF(%LN%,%UCU01%,%U%) %</pre> If none of the values is set, the result is no values. To enter a constant string in an argument, enclose it in single quotes.

Built-in Rule Function	Description
INDEX	Returns one value of a multivalued attribute. Index 1 is the first value. If the index is greater than the number of values, the result is the unset (empty) value. The following rules are equivalent to the following: <pre> \$\$\$INDEX(%*UCU01%,1)% \$\$\$FIRSTOF(%*UCU01%)% </pre>
NOTEMPTY	Returns the single value of its one argument, but reports a failure if this attribute value is not set. Example 1: Fail the account creation or update if the global user does not have an assigned UID attribute: <pre> \$\$\$NOTEMPTY(%UID%)% </pre> Example 2: Use the first name, unless it is not set, in which case use the last name. If neither is set, fail the account creation or update. <pre> \$\$\$NOTEMPTY(\$\$\$FIRSTOF(%UF%, %UL%)%)% </pre>
PRIMARYEMAIL	Returns the primary email address extracted from the multiple email addresses. The expression %UE% is equivalent to the following: <pre> \$\$\$PRIMARYEMAIL(%UEP%)% </pre>

Built-in Rule Function	Description
RDNVALUE	Treats the attribute value as an LDAP distinguished name and extracts the common name of the object from that DN: <code>\$\$\$RDNVALUE(%#eTRoleDN%)</code> This returns the common names of all assigned provisioning roles. If the user belongs to two provisioning roles with the same common name, that role name is listed once.
TOLOWER	Converts uppercase text to lowercase: <code>\$\$\$TOLOWER(%AC%)</code>
TOUPPER	Converts lowercase text to uppercase: <code>\$\$\$TOUPPER(%U%)</code>
TRIM	Removes leading and trailing blank characters from an attribute value. For example, “%UF %UL%” would generally create a value with a first and last name separated by a blank character. However, if the global user had an empty first name attribute, this rule would generate a value ending with a trailing blank. However, using <code>\$\$\$TRIM(%UF% %UL%)</code> ensures that no leading or trailing blank exists in the account attribute value even if one or the other of First Name and Last Name was unset.

Program Exit Rule Functions

You can write your own custom rule functions as program exits and reference them in rule expressions using the following syntax:

```
%[*]$function(arg[,...])[:offset,length]%
```

This is the same syntax as the built-in functions, except there is only one dollar sign (\$) here. Built-in functions are preceded by two dollar signs (\$\$). The specified *function* is the name of a registered program exit that is invoked in the domain of the endpoint where the account is created or updated.

In general, a function program exit accepts multiple multivalued or single-valued parameters and returns one multivalued or single-valued result. A program exit may return a failure that prevents the account from being created or updated. For more information about writing program exits, see the *Programming Guide for Provisioning*. Ask the author of your program exits for information about what parameters the program exit expects and what results will occur.

Account Template Management

Identity Manager lets you make changes to Account Templates at any time. When you make those changes, you are given the option to apply those changes to the associated accounts. If you decline that option, only the account template is updated. The values will be used in the future when accounts are created or updated from that account template.

However, if you accept the option of updating accounts, Identity Manager updates account attributes using the following methods:

- If you change an initial attribute, the change is propagated automatically to the accounts. This includes evaluating any rule variable in the account template attribute value and updating the account attribute if the account attribute value does not match the result of that rule evaluation.
- If you change the value of a capability attribute, the corresponding account attribute is updated, if necessary, to be in synchronization with the account template attribute value. See the description of weak and strong synchronization.
- Certain account attributes are designated by the connector as not being updated on account template changes. Examples are certain attributes that the endpoint type only allows to be set during account creation, and the Password attribute, where you would not want to accidentally reset all user's passwords with an inadvertent Yes response.

After you make a change to an account template, Identity Manager displays the Account Template Attribute Changes dialog. This dialog lists the account attributes that have changed and the account attributes that will be updated if you answer Yes to the question to update account attributes now. The dialog distinguishes the three kinds of attributes described previously with regard to how they affect accounts (initial attribute, capability attribute, template-only attribute).

Note: If you update attributes that affect the account template only, the dialog does not open. Moreover, in some endpoint types, fields on the property sheet do not correspond with the account attributes on a one-to-one basis; therefore, two or more fields might affect a single attribute.

Propagate Initial Attributes

Changing initial attributes lets you propagate changes to all accounts associated with the account template. This is an easy way to apply the same change to all accounts.

Important! You must be careful when updating initial attributes. For example, if an account is associated with multiple account templates, a change propagated from one account template might undo a change propagated earlier from another account template.

How Capability Attributes are Synchronized

When you change capability attributes in an account template, you have the choice to apply that change to the associated accounts. If you apply the change, the corresponding attribute on the accounts uses the account template value.

The affect on a particular account's attributes depends on whether the account template is defined to use weak or strong synchronization and whether the account belongs to multiple account templates.

Weak Synchronization

To specify strong or weak synchronization, click the Account Template tab on the property sheet and select or clear the Strong Synchronization check box.

Weak synchronization ensures that global users have the minimum capability attributes that their accounts should possess. Weak synchronization is the default in most endpoint types. If you update a template that uses weak synchronization and accept the option to update associated accounts, Identity Manager updates capability attributes as follows:

- If a number field is updated in an account template and the new number is greater than the number in the account, Identity Manager changes the value in the account to match the new number.
- If a check box was not selected in an account template and you subsequently select it, Identity Manager updates the check box on any account where the check box is not selected.
- If a list is changed in an account template, Identity Manager updates all accounts to include any value from the new list that was not included in the account's list of values.

If an account belongs to other account templates (whether those templates use weak or strong synchronization), Identity Manager consults only the template that is changing. This is more efficient than checking every account template. Because weak synchronization only adds capabilities to accounts, it generally is not necessary to consult those other account templates.

Note: When propagating from a weak synchronization account template, changes that would remove or lower capabilities could leave some accounts unsynchronized. Remember that with weak synchronization, capabilities are never removed or lowered. Without consulting other templates for an account, the propagation does not consider if weak synchronization is sufficient. A subsequent Synchronize Account with Account Templates might be needed to properly synchronize an account with its account templates.

Strong Synchronization

Strong synchronization ensures that accounts have the exact account attributes as those specified in the account template.

For example, suppose you add a group to an existing UNIX account template. Originally, the account template made accounts members of the Staff group. Now, you want to make the accounts members of both the Staff and System groups. All accounts associated with the account template are considered synchronized when each account is a member of the Staff and System groups (and no other groups). Any account not in the Staff group is added to both groups.

Some other factors to consider include the following:

- If the account template uses strong synchronization, any account belonging to groups, other than Staff and System, are removed from those extra groups.
- If the account template uses weak synchronization, the accounts are added to the Staff and System groups. Any account that has additional groups defined to it remains a member of these groups.

Note: If you do not apply the changes to accounts each time you update the capability attributes of an account template, you should synchronize your accounts with their account templates regularly to ensure the accounts stay synchronized with their account templates.

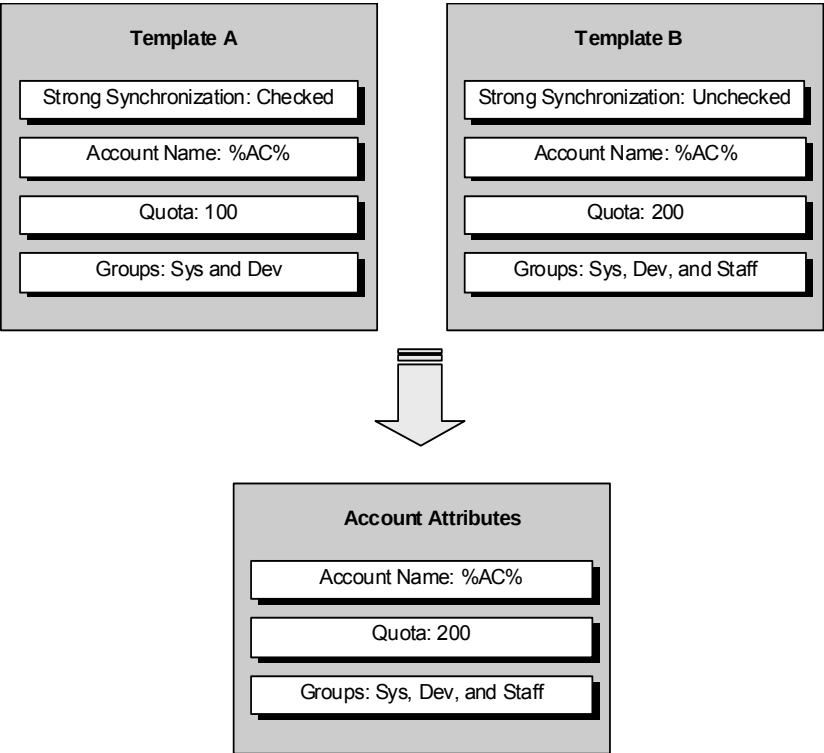
Accounts with Multiple Templates

Synchronization also depends on whether the account belongs to more than one account template. If an account has only one account template and that template uses strong synchronization, each attribute is updated to exactly match what the account template attribute value evaluates to. The result is the same as if the attribute were an initial attribute.

An account may belong to multiple Account Templates, as would be the case if a global user belonged to multiple provisioning roles each of which prescribed some level of access on the same managed endpoint. When this happens, Identity Manager combines those account templates into one effective account template that prescribes the superset of the capabilities from the individual account templates. This account template is itself considered to use weak synchronization if all its individual account templates are weak or strong synchronization if any of the individual account templates is strong.

Note: It is typical that you would use only weak synchronization or only strong synchronization for the account templates controlling one account, depending on whether your company's roles completely define the accesses your users need. If your users do not fit into clear roles and you need the flexibility to grant additional capabilities to your user's accounts, use weak synchronization. If you can define roles to exactly specify the accesses your users need, use strong synchronization.

The following example demonstrates how multiple account templates are combined into a single effective account template. In this example, one account template is marked for weak synchronization and the other for strong synchronization. Therefore, the effective account template created by combining the two account templates is treated as a strong synchronization account template. The integer Quota attribute takes on the larger value from the two account templates, and the multivalued Groups attribute takes on the union of values from the two policies.



Attributes Only for New Accounts

In an account template, certain attributes are only applied when creating the account. When you modify these attributes, the dialog informs you that the changes only affect new accounts.

For example, the Password attribute is a rule expression that defines the password for new accounts. This rule expression never updates the password of an account. Changes to the password rule expression only affect accounts created after the rule expression was set.

Similarly, a template rule expression for a read-only account attribute only affects accounts created after the rule expression was set and changes to that rule expression have no effect on existing accounts.

Recompute Template Rules

If you change the template rule expressions for an initial attribute, you have one chance to apply that change to the accounts. If you decline that option, the value is saved in the template for use when new accounts are created from the template. Running the Synchronize Accounts with Templates operation on the template later only updates capability attributes, not the attributes on the accounts.

To force a rule expression, including a constant template account attribute value, so the attribute on all the template's accounts is updated, do the following:

- Make a change to the template (for example, clear the attribute value) and decline to update all accounts.
- Change the template again by restoring the original value and accept the offer to update all accounts.

You can also use this method for Global-User-to-Account propagation. If you do not propagate the change to accounts when you change a global user attribute, you can do it again later by making two changes and accepting the offer to update accounts on the second change.

etaultil and Account Templates

You can use the Batch Utility (etaultil) to change an account template, report accounts that do not comply with the account templates from which they were created, and synchronize accounts with their account templates. This ability ensures that global users have only the privileges that they need.

Change an account Template

Use the UPDATE control statement to change the attributes of an account template. To propagate those changes to each account assigned to the account template, include the eTSyncAccounts=1 parameter.

Locate Non-Compliant Accounts

Use the REPORT control statement to check User Synchronization and Account Synchronization as follows:

- For global users, roles, account templates, and accounts, you can report all accounts that do not comply with the account templates to which they are assigned.
- For global users and roles, you can report all accounts that are not assigned to the required account template and all missing accounts (accounts required for a user by a role but that have not been created) and all the extra accounts and account template assignments.

Note: In addition to reporting accounts that do not comply with the account templates from which they were created, you can use the etautil UPDATE control statement to synchronize those accounts, bringing them into compliance with their account templates.

Synchronize Accounts with Account Templates

The following are the types of synchronization that you can perform with UPDATE control statement:

- Account synchronization for global users, roles, account templates, and accounts. Account synchronization updates account attributes to comply with the attribute values specified in the account template or account templates to which the account has already been assigned.
- User synchronization for global users and roles. User synchronization creates missing accounts (accounts that are required for a user by a role but have not yet been created) or associates missing account templates to the accounts. User synchronization also deletes extra accounts or removes extra account templates associated with the account.

Chapter 6: Provisioning Roles

This section contains the following topics:

[Provisioning Roles Management](#) (see page 67)

[Provisioning Role Performance](#) (see page 72)

[Provisioning Roles for Existing Environments](#) (see page 74)

Provisioning Roles Management

In the User Console, you can create and manage provisioning roles by choosing Roles and Tasks and selecting a task under Provisioning Roles. Tasks exist for the standard operations, such as making a user a member of a role and modifying or deleting a role.

You can also import roles that were created in the Provisioning Manager or eTrust Admin. However, Identity Manager does not support nested roles that were created in eTrust Admin.

Create a Provisioning Role

You create a provisioning role once you decide about the role requirements:

- Which Identity Management environment has users who need other accounts
- Which accounts will be associated with the role
- Who will be the members, administrators, and owners of the role

To create a provisioning role

1. In the User Console, choose Roles and Tasks, Provisioning Roles, Create Provisioning Role.

For details on each tab, click the Help link on the screen.

2. Complete the Profile tab. Only the Name field is required.

Note: You can specify custom attributes on the Profile tab that specify additional information about provisioning roles. You can use this additional information to facilitate role searches in environments that include a significant number of roles. For more information, see the *Administration Guide*.

3. Complete the Account Templates tab.
 - a. Choose an Endpoint Type, such as an ActiveDirectory.
 - b. Choose an account template. The account templates are defined in the Provisioning Manager.

The templates that you can choose are based on Endpoint Type.

- c. Add more account templates as needed for different endpoint types.

Profile **Account Templates** **Administrators** **Owners**

Select Account Template

Search for an account template
Endpoint Type ActiveDirectory

where + - = * - +

Search Clear

☐	▲ Name	Description	Comments	Endpoint
☒	ADSAccountPolicy	Default ActiveDirectory Account Template		
☐	ADSContactPolicy	Default Template for ActiveDirectory Contacts		

4. Complete the Administrators tab by adding admin rules that control who manages members and administrators of this role.
5. Complete the Owners tab by adding owner rules that control who can modify this role.
6. Click Submit.
7. To check that the role was created, choose Provisioning Roles, View Provisioning Role, then select the name of the role.

Import a Provisioning Role

Although you manage provisioning roles in Identity Manager, some provisioning roles may have been created in Provisioning Manager or an external application. For any provisioning role created outside of Identity Manager, you can reset the role owner to be an Identity Manager administrator, so you can manage it in Identity Manager.

To import a provisioning role

1. Log into the Identity Manager Environment as the user with the System Manager role.
2. Choose Roles and Tasks.
3. Choose Provisioning Roles, Reset Provisioning Role Owners and select a provisioning role created in Provisioning Manager.
4. Complete the Owners tab by adding owner rules that control who can modify this role.
5. Click Submit.

The role can now be modified, assigned, or viewed by using tasks in the Provisioning Roles category.

Order Provisioning Role Event Processing

Some default Identity Manager tasks include *events*, actions that Identity Manager performs to complete a task, that determine provisioning role membership. For example, the default Modify User task includes the `AssignProvisioningRoleEvent` and the `RevokeProvisioningRoleEvent`. Assigning or revoking a provisioning role may add or remove an account on an endpoint. In some cases, the endpoint may require that all Add actions occur before Remove actions.

To force Identity Manager to process Add actions first, you enable the Accumulation of Provisioning Role Membership Events setting in the Management Console. When this setting is enabled, Identity Manager accumulates all of the Add and Remove actions into a single event, called the `AccumulatedProvisioningRolesEvent`. For example, if the Modify User task assigns a user to three provisioning roles and removes that user from two other provisioning roles, an `AccumulatedProvisioningRolesEvent` will be generated which contains five actions: 3 Add actions and 2 remove actions.

When this event executes, all of the Add actions are combined into a single operation and sent to the Provisioning Server for processing. Once processing of the Add actions completes, Identity Manager combines the Remove actions into a single operation and sends that operation to the Provisioning Server.

Enabling this setting affects the following Identity Manager functionality:

- **Provisioning Roles Tab in User Tasks**

When an administrator adds or removes a user from a provisioning role using the Provisioning Roles tab, Identity Manager accumulates those actions into a single event.

- **Identity Policies**

All provisioning role membership events (`AssignProvisioningRoleEvent` or `RevokeProvisioningRoleEvent`) that are generated as a result of an Identity Policy evaluation are accumulated into a single `AccumulatedProvisioningRolesEvent`. Identity Manager executes this event like any other secondary event. For example, consider an identity policy set that includes two identity policies: Policy A revokes membership in the Provisioning Role A and Policy B makes users members of Provisioning Role B. If Identity Manager determines that a user no longer satisfies Policy A, but now satisfies PolicyB, an `AccumulatedProvisioningRolesEvent` that contains two actions (one for the remove action and one for the add action) is generated. The Add action is executed first and then the Remove action is executed.

- **View Submitted Tasks**

To view the status of the `AccumulatedProvisioningRolesEvent`, as well as the status for each of the individual actions, use the View Submitted Tasks task to view event details.

If one of the individual actions fails, the status of the event is failed, which moves the task to a failed state.

- **Workflow**

You can associate a workflow process with the `AccumulatedProvisioningRolesEvent`. In this case, an approver can approve or reject the entire event, which approves or rejects each of the individual events.

[Additional configuration](#) (see page 71) is required to enable workflow for individual events within the `AccumulatedProvisioningRolesEvent`.

- **Auditing**

Identity Manager audits information about the `AccumulatedProvisioningRolesEvent` and each individual event.

Note: For more information about auditing, see the *Administration Guide*.

Enable Provisioning Role Accumulation

Identity Manager provides a configuration setting in the Management Console that enables the combination of all Add and Remove actions for a provisioning role membership event into a single operation. Once combined, Identity Manager processes the Add actions as a single operation before processing the Remove actions.

This allows sequencing of events required by some endpoint types.

Note: This feature is disabled by default.

To enable Provisioning Role Membership Event Accumulation

1. Access the Identity Manager Management Console.
2. Click Environments.
3. Select the environment that you want to configure.
4. Open Advanced Settings, Provisioning.
5. Select the Enable Accumulation of Provisioning Role Membership Events check box.
6. Restart the application server.

Add Workflow Support for AccumulatedProvisioningRolesEvent

If approvals are required for the individual add/remove actions within the AccumulatedProvisioningRolesEvent, additional configuration is required for updating roles, tasks, and workflow process definitions.

Note: This additional configuration is required **only** if deployments need to approve individual actions within the AccumulatedProvisioningRolesEvent, *and* the Identity Manager environment was created in a release before Identity Manager r12 CR1.

To approve or reject individual actions within the AccumulatedProvisioningRolesEvent, an approver uses a specific approval screen that lets him select an Approve or Reject radio button for each action. If at least one action is approved, the event moves into the approved state and gets executed. If all actions are rejected, the event moves into the rejected state and then to the cancelled state.

Note: To view the status of each action, use the View Submitted Tasks task to view the details of the AccumulatedProvisioningRolesEvent.

To enable workflow for the AccumulatedProvisioningRolesEvent

1. For an existing environment, import the AccumulatedProvisioningRolesUpdate.xml in the Identity Manager Management Console, Role and Task Settings.

This file exists in the Administrative Tools in *one* of the following locations:

- **Windows:** C:\Program Files\CA\IAM Suite\Identity Manager\tools\RoleDefinitionUpdates
- **UNIX:** HOME/CA/IAM_Suite/Identity_Manager/tools/RoleDefinitionUpdates

Note: For new environments created with Identity Manager r12.0 CR1 or later, the AccumulatedProvisioningRolesUpdate.xml import is not necessary as the approval task is available with new environments.

2. Restart the application server.
3. Make sure that the Approve Accumulated Provisioning Roles task now exists by using the View Admin Task task.
4. Run the Archive.bat located in one of the following locations:
 - **Windows:** C:\Program Files\CA\IAM Suite\Identity Manager\tools\Workpoint\bin
 - **UNIX:** HOME/CA/IAM_Suite/Identity_Manager/tools/Workpoint/bin
5. Import the AccumulatedProvisioningRolesApproveProcess.zip located in one of the following locations:
 - **Windows:** C:\Program Files\CA\IAM Suite\Identity Manager\tools\WorkpointScripts
 - **UNIX:** HOME/CA/IAM_Suite/Identity_Manager/tools/Workpointscripts
6. Open Designer.bat to verify that this process definition now exists.
Identity Manager workflow now supports the AccumulatedProvisioningRolesEvent.

Provisioning Role Performance

When using Identity Manager with a Provisioning Server, there are some provisioning performance enhancements you may want to consider.

JIAM Object Cache

Identity Manager communicates with the Provisioning Server using the Java IAM (JIAM) API. To improve communication performance, you configure a cache for objects retrieved from the Provisioning Server.

Enable the JIAM Cache

To enable the JIAM Cache

1. Access the environment settings through the Management Console. Click Advanced Settings, Miscellaneous.
2. Configure the User Defined Property for the JIAM Cache.
 - **Property**—JIAMCache
 - **Value**—true
3. Click Add.
4. Click Save.

The User Defined Property is saved.

Define the JIAM Cache TTL (Time-to-live)

The JIAM Cache stores information for a specified period of time before the data expires. This period of time is referred to as time-to-live (TTL). You set the JIAM Cache TTL value (in seconds) to define how long data remains in the cache.

To gain the maximum benefit from locally cached data, you balance performance gains against timely data. We recommend a minimum TTL value of 1 day with a maximum value of 7 days. See the following table for time-to-live values to use:

Desired Lifetime	TTL Settings (secs)
24 hours (1 day)	86,400
72 hours (3 days)	259,200
120 hours (5 days)	432,000
168 hours (7 days)	604,800

To define the JIAM Cache TTL

1. Access the Environment through the Management Console. Click Advanced Settings, Miscellaneous.
2. Configure the User Defined Property for the JIAM Cache TTL.
 - **Property**—JIAMCacheTTL
 - **Value**—number of seconds that data remains in the JIAM Cache
Default: 300
3. Click Add.
4. Click Save.

The User Defined Property is saved.

Session Pooling

To improve performance, Identity Manager can pre-allocate a number of sessions to be pooled for use when communicating with the Provisioning Server.

For more information on Session Pooling, see the *Management Console Online Help*.

IAMServer Initialization

In Identity Manager implementations that include provisioning, the Java IAM (JIAM) API enables communication between the Identity Manager server and the Provisioning Server. The JIAM API establishes a server instance, called an IAMServers instance, and a server session, called an IAMSession instance.

IAMServers initialization takes place when an Identity Manager environment that is associated with a provisioning directory is initialized during application startup.

Provisioning Roles for Existing Environments

If you import custom roles definitions and want to enable provisioning on an environment, you must *also* import the Provisioning Only role definitions provided in the Identity Manager Administrative Tools. These role definitions can be found in this folder:

IdentityMinder.ear\management_console.war\WEB-INF\Template\environment

Chapter 7: Passwords in Endpoint Accounts

This section contains the following topics:

[Choices for Password Management](#) (see page 75)

[Password Synchronization](#) (see page 75)

[Pluggable Authentication Module \(PAM\)](#) (see page 82)

Choices for Password Management

The complexity of user management is forcing passwords to be more secure, yet more flexible. Identity Manager provides the following tools to give you control and flexibility when administering passwords:

Password Synchronization

Intercepts password changes made by users on their endpoint systems and propagates the changes to all the accounts associated with those users.

Note: Password Synchronization agents are available for other Identity Manager connectors. To determine if Windows password synchronization agent works with your connector, see the *Connectors Guide*.

Pluggable Authentication Module (PAM)

Authenticates passwords so users can employ their native system passwords to log on to any interface in Identity Manager.

Password Synchronization

Identity Manager is able to capture (intercept) password changes made to native Windows systems and propagate them to all accounts belonging to that user, assuming your standard is that a user's password is the same on all systems.

However, you can designate that accounts on certain endpoints are excluded from password propagations, including ones originating from psync agents. Also, you can use provisioning domain configuration parameter (Password Synchronization/Update Only Global User) to designate requests from psync agents only update the global user (and corporate user), but not any of the global user's other accounts.

Note: Password synchronization works for global users with multiple accounts, each in a different container.

When the Password Synchronization agent detects that users are changing their passwords, the agent intercepts the request and sends it to the Provisioning Server, which then propagates the new password to accounts associated with the global user.

The requirements for Identity Manager password synchronization include the following:

- The Identity Manager Password Synchronization Agent must be installed on the system on which password changes are to be intercepted.
- The system must be managed as an acquired endpoint.
- The Password synchronization agent is installed check box must be selected on the acquired endpoint's properties.
- The accounts on the managed systems must be explored and correlated to Identity Manager global users.
- The Enable Password Synchronization Agent flag is selected on the Global User property sheet's Password tab for each global user for which password synchronization is to be enabled
- The provisioning domain configuration parameter Identity Manager Server/Use External Password Policies must be selected.

Important! Use care in formulating password creation rules, so that a single password can be used on all systems. For example, if Windows passwords must be 12 characters, any system that accepts passwords only up to 10 characters will reject the change during synchronization.

The Identity Manager server is not aware of the password restrictions on the endpoint. When working with endpoint accounts, the password policy should be stricter than the password policy on the endpoints.

Install the Password Synchronization Agent

You can install the Password Synchronization Agent on any managed Windows computer where global users log on. The Agent runs in the background on these machines.

Run the Installation Program

Note the following:

- The machine where you are installing the Agent must be managed by the Provisioning Server.
- Make sure that you create Administrator named etapwsad.

To install the Password Synchronization Agent

1. Locate the Identity Manager installation DVD or other installation media.
2. Start the Media Explorer.
3. Select Install Products, Infrastructure, Password Synchronization Agent for Windows.
4. Complete the Installation Wizard as follows:

Identity Manager Password Sync Configuration (Step 1 of 3)

- a. In the Host Name field, enter the name of the Provisioning Server machine.
- b. In the LDAP Port field, enter the port number that the machine uses when it connects to the Provisioning Server.
- c. Click the Find domain button to get the Domain List.
- d. In the Domain drop-down box, select the name of the domain.
- e. Click Next.

Identity Manager Password Sync Configuration (Step 2 of 3)

- f. In the Administrator field, enter etapwsad as the default global user name for the Password Synchronization Agent. This user must be created with the PasswordAdministrator profile. It does not exist by default.
- g. In the Password Administrator field, enter the password of the Administrator.
- h. Click Next.

Identity Manager Password Sync Configuration (Step 3 of 3):

- i. In the endpoint type Name drop-down box, select the type of machine on which you are installing the Agent.
 - j. In the endpoint drop-down box, select the name of machine on which you are installing the Agent.
 - k. Click Configure.
5. Click OK when prompted to complete the installation.

Configure Alternate Servers for the Password Synchronization Agent

To configure an alternate server for the Password Synchronization Agent, you use the Password Synchronization Agent Configuration wizard.

To add alternate server for the Password Synchronization Agent

1. Run PwdSyncConfig.exe located in *password_sync_folder\bin*.
2. Enter the following configuration information:

Host

Specify the primary Provisioning Server host name.

This populates the Server URL field with the host name you specify.

LDAP port

Specify the port number that the machine uses when it connects to the Provisioning Server.

Identity Manager populates the Server URL field with the host and port that you specify.

3. Click the Find domain button to get the Domain List.
4. In the Domain drop-down box, select the name of the domain and click Next.
5. Add the host name and port of the alternate servers in the Server URLs field using the following format:

`ldaps://primaryhost:20390,ldaps://alternatehost1:20390`
6. Click Next.
7. Complete the remaining fields in the configuration wizard.

How the Password Synchronization Agent Works

The propagation process begins when global users change their password on a Windows system using any method. After the password is entered, the following occurs:

1. The Windows operating system checks to make sure the password meets its password policy. If Windows does not accept the password, the change request is rejected, an error message appears, and no further action, including synchronization, is taken.
2. The Windows system passes the password change request to the Identity Manager Password Synchronization agent, which, if configured for password quality checking, submits the password to the Provisioning Server for password quality checking. If the password does not meet the Identity Manager quality rules, the change request is rejected and an error message displays. The Windows password remains unchanged and no synchronization takes place.
3. A password that meets the quality rules of both Windows and Identity Manager is submitted by the Password Synchronization Agent to the Provisioning Server for propagation.
4. Identity Manager updates the global user password and propagates the new password to some or all accounts associated with the global user.

Note: Your password policies for Windows and Identity Manager must be identical or consistent, because the error messages displayed are based on the Windows password policy, even if Identity Manager rejects the request.

The `password_update_timeout` configuration parameter (`eta_pwdsync.conf`) specifies how long (in seconds) the PSA waits for the password-change-propagation confirmation from the Identity Manager server. If the PSA does not receive a confirmation during that time, it proceeds as if the propagation succeeded and logs a warning (`eta_pwdsync.log`) that password change propagation could not be verified. The minimum value for the parameter is zero (0), which means that the PSA will not wait for confirmation. For more information, see `eta_pwdsync.conf--Configure Password Synchronization Agent` in the Provisioning Manager help.

Account-Level Password Quality Checking

Password Quality Checking is performed when accounts on managed endpoints are created or modified or when global user passwords are set. Password Quality Checking on accounts is limited to checks based on the characters in the password. Checks on global users that are based on the history of recent changes (frequency of password update and frequency of password reuse) are not performed on accounts because Identity Manager does not intercept all password changes for account passwords. Therefore, it does not have an accurate password change history with which to perform these checks.

The checking of account passwords is controlled by the following domain configuration parameters:

- Endpoint type/Check Account Passwords
- Endpoint type/Check Empty Account Passwords

The value for each parameter specifies for each managed endpoint the level of checking that should be performed. The endpoint can be specified in the following ways:

```
ALL
-ALL
<NamespaceName>
-<NamespaceName>
<NamespaceName>:<DirectoryName>
-<NamespaceName>:<DirectoryName>
```

The forms that include a minus (-) sign, disable the parameter. The forms without it enable the parameter. The [-]<NamespaceName> forms control all endpoints of the indicated endpoint type, while the [-]<NamespaceName>:<DirectoryName> forms control individual endpoints. The [-]ALL forms control all endpoints of all endpoint types. The default value for both parameters is -ALL.

Each of these parameters can be specified many times. If multiple values specify the same endpoint, the last value is used. You can place general rules first and specific rules later to override the general rule.

The Check Account Passwords parameter provides checking equivalent to global user password quality checking. With this parameter enabled for an endpoint, Identity Manager checks any password in a requested change for an existing account, including attempts to set an empty password. During account creation, if no password is provided, password quality checking is not performed.

Check Empty Account Passwords provides the added checking of empty passwords when creating accounts. If the password profile is enabled and requires at least a single-character password, an empty password causes account creation to fail. This parameter is separate from Check Account Passwords because in some endpoint types it is acceptable to create an account with no password.

Note: Account password quality checking is skipped for synchronized account passwords if the supplied password matches the current global user password.

Password Quality Enforcement

The Password Synchronization option intercepts password changes requests on native systems (for example, Windows NT/ADS) and submits them to the Identity Manager server. The Server synchronizes the global user password and account passwords associated with the global user. Both Identity Manager password quality rules for a password profile and native system password quality rules (Windows NT/ADS) can be used to enforce password quality control.

Configure Password Synchronization

The Password Synchronization Agent is automatically configured through the Password Synchronization Configuration Wizard that appears when it is installed. There may be times when you want to set certain configuration keys for this wizard. For example, to set password quality checking on the Identity Manager server, or to change the name of the Password Synchronization log file or its location, change the configuration keys in the `eta_pwdsync.conf` file. For more information, see the `eta_pwdsync.conf` file in the Provisioning Manager help.

Failover

If a server is down or heavily loaded, the Password Synchronization agent will failover from one server to another. For example, when a server is busy with a heavy load, such as an explore/correlate operation, and cannot accept password synchronization requests, password synchronization will failover to an alternative server. Multiple servers must be configured to serve the same domain. For more information about failover, see the *High Availability Guide*.

Enable Log Messages

To know why a password modification was rejected, view the log messages sent from the Password Synchronization Agent. All logged messages are stored in the `eta_pwdsync.log` file. By default, this file is located in the `..\Program files\CA\Identity Manager Password Sync Agent` folder.

PSA logging (going into `eta_pwdsync.log` file) has the following messages:

- Error messages, which are always logged.
- Diagnostic (process flow, trace) messages, which can be enabled or disabled based on the value of the `logging_enabled=yes|no` parameter in the `eta_pwdsync.conf` file.

To better diagnose problems, review the `eta_pwdsync.log` and the Provisioning Server log for the same time period.

The previous `log_level` configuration parameter has been deprecated but left for backward compatibility: `log_level=0` translates into `logging_enabled=no` and `log_level=anything` else translates into `logging_enabled=yes`. If both old and new parameters are present in the configuration file, the explicit setting of `logging_enabled=yes|no` parameter overrides the indirect setting performed through the old `log_level=number`.

Note: The list of available Connectors was included in the `eta_pwdsync.log`, but the Agent no longer provides that information.

Verify the Installation

After the Password Synchronization Agent installation is complete, change a password on the Windows system to verify that the global user password associated with the account is changed also.

Pluggable Authentication Module (PAM)

Identity Manager provides a Pluggable Authentication Module (PAM) that allows the Provisioning Server to authenticate against external security systems, such as a Primary Domain Controller (PDC).

Note: An external security system need not be a Windows system. But if it is, you can use one of the predefined PAM modules provided with the Provisioning Server (two for Active Directory and one for Windows NT). Otherwise, you must write your own PAM module as described in the *Programming Guide for Provisioning*.

When PAM is enabled, global users can log on to any Provisioning Manager or `etaultil` using the user's password in the external security system. It is only the password check that the Provisioning Server defers to the PAM module. All additional user information, such as suspension state, Self-Administration check box, and administrative privileges, that controls what actions you may perform still reside in the provisioning directory as properties of the global user or related objects.

How PAM Works

The PAM authenticates passwords using the following process:

1. Global users log in to an Identity Manager interface with their native system passwords.
2. Identity Manager sends the authentication information to the PAM loader.
3. The PAM loader sends the authentication information to PAM where it is authenticated against the system. The Identity Manager server grants or fails login requests based on the results returned from the PAM DLL.

Sample PAM Programs

Identity Manager provides the following sample PAM programs:

- Windows NT
- ADS
- ADSDomain
- Generic PAM sample

Note: The Generic PAM sample is included in \Provisioning SDK\admin\samples\SDK\PAM.

Each sample program includes the etaPAM.dll binary, which lets you authenticate to a local Windows system. Sample source code is provided in the Provisioning SDK for reference purposes only. Customization may be needed for a production deployment.

Any global user who needs to authenticate to Identity Manager must exist in the Windows domain. This includes the special global user, etapwsad. This user must be created with the PasswordAdministrator profile. It does not exist by default.

If these users do not exist, the Identity Manager clients will not function.

The Generic PAM Sample shows an example implementation of the PAM interface that does not actually communicate with an authentication system. This sample demonstrates how to implement the PAM interface on Windows or Solaris, without being tied to a particular platform for carrying out the user authentication.

Use PAM

The PAM program is called etaPam.dll. The etaPam.dll is in these folders:

- PSHOME/PAM/ADS
- PSHOME/PAM/ADSMultiDomain
- PSHOME/PAM/NT

The ADS version authenticates to a Windows domain controller in a single-domain or multi-domain ADS environment. To ensure that ADS domains are trusted and can be reached, the Provisioning Server must be in the ADS domain. The NT version authenticates to the local accounts database. By default, these programs authenticate passwords on Windows systems only.

To use PAM with other systems, write another PAM program for the system. Sample source for the Windows or UNIX PAM modules provided with Identity Manager can be obtained by installing the Provisioning SDK. The generic PAM runs on UNIX and the NT and ADS samples run on Windows.

Using PAM and Password Expiration

If your environment is configured for PAM, you must inform PAM through the etapam_id.conf configuration file if the endpoint PAM is to consult has been acquired in Identity Manager as a managed endpoint. This allows Identity Manager to update a user's account password on this managed endpoint every time users or administrators update their global user passwords. For more information, see Update Global User Passwords.

If you configure PAM to consult an endpoint that has not been acquired in Identity Manager as a managed endpoint, disable the password expiration features. If you do not disable the password expiration feature, your users will be confused as they are asked to change their global user password although that password is not used for authentication.

To disable password expiration

1. Click the System Task button in the Provisioning Manager.
2. Click Password Profile.
3. Click the Expiration/Locking tab.
4. Clear the Expiration Period and click Apply.

Update Global User Passwords

You can configure PAM so that any update to a global user password also updates the account in the external security system that PAM uses for password verification. To do this, the external security system needs to be an acquired endpoint in Identity Manager, that is, a managed endpoint.

To configure automatic updates to external security systems, perform the following steps:

1. Add the following lines to your `etapam_id.conf` file:

- For Active Directory:

```
endpoint-type=ActiveDirectory
endpoint-domain=YOUR_DOMAIN
endpoint-name=YOUR_DIRECTORY_NAME
```

- For Windows NT:

```
endpoint-type=Windows NT
endpoint-domain=YOUR_DOMAIN
endpoint-name=YOUR_DIRECTORY_NAME
```

2. Replace `YOUR_DOMAIN` and `YOUR_DIRECTORY_NAME` with the domain and endpoint names that identify this managed endpoint in Identity Manager. You can omit `endpoint-domain` if the endpoint is in the server's local domain.
3. Start or restart the Provisioning Server service to have this change take effect.

When a managed endpoint is enabled in this way, any change to a global user password is also applied to the password of the matching account on the indicated endpoint if that account is correlated to the global user. The account password update occurs whether or not password propagation to accounts is requested. It occurs even in cases where password propagation would not have occurred. For example, the update occurs even if password propagation to the endpoint has been disabled or if the global user is marked as restricted. If password propagation to accounts is requested, the global user's other accounts are updated as well.

For Active Directory, a matching account is one whose `samAccountName` attribute is equal to the global user's name. For Windows NT, a matching account is one whose account name is the same as the global user's name.

Use PAM with Multiple ADS Domains

PAM must be configured on each Provisioning Server individually. Each PAM module only handles the authentication for the Provisioning Server where it is installed.

PAM allows a comma-delimited list of ADS domains to be added to the `etapam_id.conf` file.

To use PAM with multiple ADS domains

1. Ensure that the user account, specified as the user the Provisioning Server service logs on with, has the Act as Part of the Operating System privilege. If you need to add this privilege to the user, you must restart this service.
2. Ensure that `enable=yes` appears in the `etapam_id.conf` file.

3. Add a comma-delimited list of ADS domains to be added to the `domain=` setting in the `etapam_id.conf` file. These domains must be trusted by the domain in which Provisioning Server is installed, or be its own local domain. The following is an example:

```
domain=LocalDomain,Trusted1,Trusted2
```

Identity Manager attempts to authenticate to each listed domain until it is successful.

Copy `etapam_id.conf` from the `PSHOME\PAM\ADSMultiDomain` directory to `PSHOME\PAM`.

4. Copy `etapam.dll` from the `PSHOME\PAM\ADSMultiDomain` directory to `PSHOME\bin`.
5. Restart the Provisioning Server service to make the Provisioning Server aware of any changes that were made to `etapam_id.conf`.
6. Log in with a user name that is both an Identity Manager Global User name and an ADS account name in one of the ADS domains specified in `etapam_id.conf`. Use the account's ADS password, not its Global User password. The Global User must have the Provisioning Server administrative privileges that are necessary for the user's purpose.

Troubleshooting PAM

The Provisioning Server Trace log (regardless of the configured log level), upon start up of the Provisioning Server service, will contain information about whether PAM is enabled or disabled and whether there is a valid configured PAM managed endpoint. To see this information, view the `PSHOME\Logs\etatransYYYYMMDD.log` and look for lines containing the PAM: string.

Common messages include the following:

PAM: Initialization started

Signals the start of PAM processing.

PAM: Not enabled

Indicates that PAM is not being used and could mean any of the following:

- The etapam_id.conf file was not found or invalid.
- The etapam_id.conf file was found but the enabled parameter was not set to yes.
- The etapam.dll file was not found on the execution search path or that the supplied etapam.dll file could not be successfully loaded.

PAM: No PAM managed endpoint

Indicates that no managed endpoint was specified using endpoint-type, endpoint-domain and endpoint-name parameters in etapam_id.conf.

PAM: Missing EPTYPE or EPNAME

PAM: Unable to find specified domain

PAM: Unable to find specified endpoint

These messages indicate that there was a problem identifying the managed endpoint using the supplied endpoint-type, endpoint-domain, and endpoint-name parameters in etapam_id.conf.

PAM: Managed endpoint configured

Indicates that the managed endpoint identified by endpoint-type, endpoint-domain, and endpoint-name was valid. The next line logged is the full LDAP distinguished name of the managed endpoint.

Activate System PAM Debug Mode

The PAM library can provide debug information during execution. After enabling the system to collect debug output, you can use the gathered information to track PAM-API invocations and determine failure points in the current PAM setup. To enable PAM debug output, create an empty /etc/pam_debug file. The PAM library checks for existence of the /etc/pam_debug file and if found, enables syslog output.

Configuration

Configuration of the UNIX Password Synchronization feature involves setting parameters in the following files:

- `/etc/pam_CA_eta/pam_CA_eta_conf`
- `/etc/pam.conf`

Important! Because the password of a highly-privileged Identity Manager user is stored in the `pam_CA_eta.conf` configuration file, that file must be readable only by the root account. Note that the file settings in the package file include `owner=root` and `mode=500` and that the `-p` switch of the `cp` command preserved them during installation.

Configure the `pam_CA_eta.conf` File

1. Navigate to the `/etc/pam_CA_eta` folder.
2. Edit the `pam_CA_eta.conf` file. This configuration file contains its own documentation.

If you want UNIX Password Synchronization to be able to use an alternate server when the primary server is unavailable, substitute the host name of the alternate server for `ALT_SERVER`. Otherwise remove `[ALT_SERVER]` entry.

```
#
# Computer Associates - Identity Manager
#
# pam_CA_eta.conf
#
# Configuration file for the Unix PAM password module "pam_CA_eta"
#
# keyword: server
# description: the Identity Manager LDAP primary and optional alternate
server hostnames # value: a valid hostname and an optional alternate server #
default: no default server ETA_SERVER ALT_SERVER

# keyword: port
# description: the numeric TCP/IP port number of the Identity Manager
LDAP server
# value: a valid TCP/IP port number
# default: 20390
# port 20390
```

```
# keyword: use-tls
# description: does it use the secured LDAP over TLS protocol ?
# value: yes or no
# default: yes
# use-tls yes
```

```
# keyword: time-limit
# description: the maximum time in seconds to wait for the end of an
LDAP operation.
# value: a numeric value of seconds
# default: 300
# time-limit 300
```

```
# keyword: size-limit
# description: the maximum number of entries returned by the Identity
Manager server
# value: a numeric value
# default: 100
# size-limit 100
```

```
# keyword: root
# description: the root DN of the Identity Manager server
# value: a valid DN string
# default: dc=eta
# root dc=eta
```

```
# keyword: domain
# description: the name of the Identity Manager domain
# value: a string
# default: no default
domain    DOMAIN
```

```
# keyword: user
# description: the Identity Manager Global User name used to bind to the
Identity Manager server
# value: a valid Global User name string
# default: etaadmin
# user etaadmin
```

```
# keyword: password
# description: the clear-text password of the "binding" Identity Manager
Global User
# value: the password of the above Global User
# default: no default
password SECRET

# keyword: directory-type
# description: the Identity Manager Unix Directory type of this Unix server
# value: ETC or NIS
# default: ETC
# directory-type ETC

# keyword: directory-name
# description: the Identity Manager Unix Directory name of this Unix
server
# value: a valid Unix Directory name string
# default:
#   ETC: the result of the "hostname" command (ie: gethostname() system
call)
#   NIS: "domain [hostname]" where "domain" is the result of the
"domainname" command
#   (ie: getdomainname() system call) and "hostname" the result of the
"hostname"
#   command (ie: gethostname() system call)
# directory-name dirname

# keyword: tls-cacert-file
# description: the name of the Identity Manager CA certificate file
# value: a valid full path file name
# default: /etc/pam_CA_eta/eta2_cacert.pem
# tls-cacert-file /etc/pam_CA_eta/eta2_cacert.pem

# keyword: tls-cert-file
# description: the name of the Identity Manager client certificate file
# value: a valid full path file name
# default: /etc/pam_CA_eta/eta2_clientcert.pem
# tls-cert-file /etc/pam_CA_eta/eta2_clientcert.pem

# keyword: tls-key-file
# description: the name of the Identity Manager client private key file
# value: a valid full path file name
# default: /etc/pam_CA_eta/eta2_clientkey.pem
# tls-key-file /etc/pam_CA_eta/eta2_clientkey.pem
```

```
# keyword: tls-random-file
# description: the name of the "pseudo random number generator" seed
file
# value: a valid full path file name
# default: /etc/pam_CA_eta/prng_seed
# tls-random-file /etc/pam_CA_eta/prng_seed
```

```
# keyword: use-status
# description: this module will exit with a non-zero status code in case of
failure.
# value: yes or no
# default: no
# use-status no
```

```
# keyword: verbose
# description: this module will display informational or error messages to
the user.
# value: yes or no
# default: yes
# verbose yes
```

Note: The parameters "server," "domain," and "password" do not have a default value and need to be updated.

In a multi-domain environment, the server must be the host name of the Identity Manager server for the domain where the UNIX endpoint is acquired. The root DN must be the DN of the root Identity Manager server. The user must have administrative privileges on the child domain, such as DomainAdministrator Profile and UserAdministrator Profile, for all the domains where global users associated to accounts on that UNIX endpoint are located.

Configure the pam.conf File

The /etc/pam.conf file is the main PAM configuration file. You must edit the file to insert a line in the password service stack. On some Linux systems, the pam.conf file is replaced with /etc/pam.d, so you will need to edit the /etc/pam.d/system-auth file.

1. Navigate to the /etc directory, or /etc/pam.d directory if you are configuring the PAM module on an appropriate Linux system.
2. Edit the pam.conf file to insert an eTrust Password Synchronization line in the password service stack. For platform-specific configurations, see the examples that follow:

```
passwd password required /usr/lib/security/pam_unix.so
passwd password optional /usr/lib/security/pam_CA_eta.so
```

3. (Optional) You can add the following optional parameters on the `pam_CA_eta` module line:

config = /path /file

Indicates the location of an alternate configuration file.

syslog

Sends error and informational messages to the local syslog service.

trace

Generates a trace file for each password update operation. The trace files are named `/tmp/pam_CA_eta-trace.<nnnn>` where `<nnnn>` is the PID number of the `passwd` process.

4. Implement the following platform-specific configuration changes:

For AIX systems, add the following lines at the bottom of the `/etc/pam.conf` file:

```
#
# Identity Manager Unix Password Synchronization
#
login    password optional    /usr/lib/security/pam_CA_eta.so syslog
passwd   password optional    /usr/lib/security/pam_CA_eta.so syslog
rlogin   password optional    /usr/lib/security/pam_CA_eta.so syslog
su        password optional    /usr/lib/security/pam_CA_eta.so syslog
telnet    password optional    /usr/lib/security/pam_CA_eta.so syslog
sshd      password optional    /usr/lib/security/pam_CA_eta.so syslog
OTHER     password optional    /usr/lib/security/pam_CA_eta.so syslog
```

For HP-UX systems, add the following lines at the bottom of the `/etc/pam.conf` file:

```
#
# Identity Manager Unix Password Synchronization
#
login    password optional    /usr/lib/security/libpam_CA_eta.1 syslog
passwd   password optional    /usr/lib/security/libpam_CA_eta.1 syslog
dtlogin   password optional    /usr/lib/security/libpam_CA_eta.1 syslog
dtaction  password optional    /usr/lib/security/libpam_CA_eta.1 syslog
OTHER     password optional    /usr/lib/security/libpam_CA_eta.1 syslog
```

For HP-UX Itanium2, add the following lines at the bottom of the `/etc/pam.conf` file:

```
#
# Identity Manager Unix Password Synchronization
login    password optional    /usr/lib/security/$ISA/libpam_CA_eta.1
syslog
passwd   password optional    /usr/lib/security/$ISA/libpam_CA_eta.1 syslog
dtlogin   password optional    /usr/lib/security/$ISA/libpam_CA_eta.1 syslog
dtaction  password optional    /usr/lib/security/$ISA/libpam_CA_eta.1 syslog
OTHER     password optional    /usr/lib/security/$ISA/libpam_CA_eta.1
syslog
```

For Sun Solaris systems, add the `pam_CA_eta` line after the existing `pam_unix` line:

```
#
# Password management
#
other password required      /usr/lib/security/pam_unix.so.1
other password optional      /usr/lib/security/pam_CA_eta.so syslog
```

For Linux systems, add the `pam_CA_eta` line between the existing `pam_cracklib` and `pam_unix` lines:

```
password required      /lib/security/pam_cracklib.so retry=3 type=
password optional      /lib/security/pam_CA_eta.so syslog
password sufficient     /lib/security/pam_unix.so nullok use_authtok md5
shadow
password required      /lib/security/pam_deny.so
```

5. For AIX systems, edit the `/etc/security/login.cfg` file to set `auth_type = PAM_AUTH`. This enables the PAM framework, which is not enabled by default. This is a run-time setting so you do not have to reboot the system for it to take effect.

UNIX Password Synchronization and PAM

Identity Manager also provides a password synchronization module that detects password change events through the UNIX PAM framework. The UNIX Password Synchronization module notifies the Provisioning Server of a password change. The Provisioning Server then finds the associated global user and updates endpoint accounts for that user.

Prerequisites

UNIX operating systems that support the PAM framework include:

- AIX v5.3 on Power platform with PAM enabled
- HP-UX v11.00 on a PA-RISC platform, and Itanium® 2 platforms
- Solaris v2.6 and higher on Sparc and Intel platforms
- Linux with glibc v2.2 and higher on s390 or Intel i386 platform

How UNIX PAM Works

The following process describes the UNIX PAM feature's functions:

1. A UNIX user decides to change his password, or is forced to do so by system settings or manual intervention, or the user's password is changed by an administrator.
2. The new password is submitted to the PAM framework password service.

3. The PAM framework's password service invokes the PAM library to update the local UNIX security files.
4. The PAM framework's password service invokes the Provisioning Server UNIX password synchronization module (pam_CA_eta) to notify the Provisioning Server of the password change.
5. The Provisioning Server updates the password of the associated Global User and all accounts associated with the Global User.

Requirements

The following are requirements for using the UNIX Password Synchronization feature:

- The UNIX Password Synchronization agent must be installed on the UNIX system on which you want to detect password changes.
- The Provisioning Server UNIX Remote agent and CAM must be installed on the UNIX system on which the UNIX Password Synchronization agent resides.
- The system must be managed as an acquired endpoint.
- The Password Synchronization agent is installed check box must be selected on the acquired endpoint's properties.
- The accounts on the managed systems must be explored and correlated to Provisioning Server global users.
- The Enable Password Synchronization Agent flag is selected on the Global User property sheet's Password tab.

Installation

To install the UNIX PAM feature, follow these steps:

1. Choose the package file that corresponds to your UNIX platform:

UNIX operating system	Package file name
HP-UX v11 PA-RISC	pam_CA_eta-1.1.HPUX.tar.Z
HP-UX Itanium2	pam_CA_eta1.1HPUX-IA64.tar.Z
AIX v5.3 Power	pam_CA_eta-1.1.AIX.tar.Z
Solaris Sparc	pam_CA_eta-1.1.Solaris.tar.Z
Solaris Intel	pam_CA_eta-1.1.SolarisIntel.tar.Z
Linux x86	pam_CA_eta-1.1.Linux.tar.gz
Linux s390	pam_CA_eta-1.1.LinuxS390.tar.gz

2. Transfer the chosen package file to a temporary folder (/tmp) on the UNIX server using FTP in binary mode, or any other file transfer tool that supports binary files. A sample transfer session might appear as follows:

```
W:\Pam>ftp user01
Connected to user01.company.com.
 220 user01 FTP server (Version 1.2.3.4) ready.
 User (user01.company.com:(none)): root
331 Password required for root.
 Password:
 230 User root logged in.
ftp> cd /tmp
250 CWD command successful.
ftp> bin
200 Type set to I.
ftp> put pam_CA_eta-1.1.HPUX.tar.Z
200 PORT command successful.
 150 Opening BINARY mode data connection for pam_CA_eta-1.1.HPUX.tar.Z.
 226 Transfer complete.
ftp: 117562 bytes sent in 0,09Seconds 1306,24Kbytes/sec.
ftp> quit
```

3. Logon as the root user on the UNIX server and extract the package file:

```
# cd /tmp
# zcat pam_CA_eta-1.1.<platform>.tar.Z | tar -xf -
```

On Linux, use the command:

```
# tar -xzf pam_CA_eta-1.1.<platform-hardware>.tar.gz
```

4. Copy the configuration and TLS files to the default configuration folder:

```
# cd pam_CA_eta-1.1
# mv pam_CA_eta /etc
```

5. Copy the pam_CA_eta module to the Security libraries folder:

On AIX, use the command:

```
# cp -p pam_CA_eta.o /usr/lib/security/
```

On HP-UX, use the command:

```
# cp -p libpam_CA_eta.1 /usr/lib/security/
```

On HP-UX Itanium2, use the command:

```
# cp -p libpam_CA_eta.1 /usr/lib/security/hpux32
```

On Linux i386 or s390, use the command:

```
# cp -p pam_CA_eta.so /lib/security/
```

On Solaris Sparc or Intel, use the command:

```
# cp -p pam_CA_eta.so /usr/lib/security/
```

6. (Optional) Copy the Testing programs (for more details, see Troubleshooting UNIX Password Synchronization):

```
# cp -p test_* /etc/pam_CA_eta
# cp -p pam_test* (/usr)/lib/security/
```

Troubleshooting UNIX Password Synchronization

You can troubleshoot the UNIX PAM feature using syslog and trace messages, and by testing the configuration, LDAP/TLS connection, the password synchronization, and the PAM framework.

Activate Syslog Messages

Add the syslog parameter to the pam_CA_eta line in the /etc/pam.conf file to let the pam_CA_eta module generate informational and error messages. When the logging option is in use, the UNIX administrator sees information messages in the syslog files each time a UNIX account changes its password. These messages should provide enough information to diagnose basic problems.

You could set this option permanently on production systems as it does not require many more resources than when running in silent service.

Activate Trace Messages

If the syslog messages do not provide enough information, the trace mode can provide more details. For each password update operation, the trace module generates a file named /tmp/pam_CA_eta-trace.<nnnn> (where <nnnn> is the PID of the *passwd* process) with an entry for most of the function calls used by the module and the data used or returned by those functions.

Even though the trace files are only readable by the root account, they will contain the clear-text new passwords. For this reason, this parameter should *not* be used permanently on a production system.

Test the Configuration File

You can use the test_config tool, located in the /etc/pam_CA_eta directory, to verify the configuration file. A sample command line entry follows:

```
/etc/pam_CA_eta/test_config [config=/path/to/config_file]
```

An example session follows:

```
./test_config [config=/path/to/config_file]
# ./test_config
./test_config: succeeded
Trace file is /tmp/test_config-trace.1274
```

As the command output shows, a trace file was generated which contains all the details of the configuration file parsing.

Test the LDAP/TLS Connection

You can use the `test_ldap` tool, located in the `/etc/pam_CA_eta` directory, to verify the connection to the Provisioning Server (using the configuration file parameters). A sample command line entry follows:

```
/etc/pam_CA_eta/test_ldap [config=/path/to/config_file]
```

An example session follows:

```
./test_ldap [config=/path/to/config_file]
# ./test_ldap: succeeded
Trace file is /tmp/test_ldap-trace.1277
```

As the command output shows, a trace file was generated which contains all the details of the configuration file parsing and the connection to the Provisioning Server.

Test the Password Synchronization

You can use the `test_sync` tool, located in the `/etc/pam_CA_eta` folder, to verify that the password update of a local account is effectively propagated by the Identity Manager server. A sample command line entry follows:

```
/etc/pam_CA_eta/test_sync <user> <password> [config=/path/to/config_file]
```

An example session follows:

```
# /etc/pam_CA_eta/test_sync pam002 newpass1234
Identity Manager password synchronization started.
:ETA_S_0245<MGU>, Global User 'pam002' and associated account passwords updated
successfully: (accounts updated: 2, unchanged: 0, failures: 0)
Identity Manager password synchronization succeeded.
/etc/pam_CA_eta/test_sync: succeeded
Trace file is /tmp/test_sync-trace.2244
```

As the command output shows, a trace file was generated which contains all of the details of the configuration file parsing, the connection to the Provisioning Server, and the update of the account.

When using the verbose mode (by using the default verbose yes parameter in the configuration file), the command provides informational and potential error messages about the password propagation.

Test the PAM Framework

A PAM test library is available to verify that the password changes are correctly detected by the PAM framework. To use the library, follow these steps:

1. Copy the `pam_test` file to the `/usr/lib/security(/hpux32)` folder.
2. Edit the `/etc/pam.conf` file to add `pam_test` to it.
3. Issue a `passwd` command on a test user and then search for the `pam_test[<pid>]` tagged line in the `syslog` file.

The command output shows the name of the generated trace file, similar to the following:

```
pam_test[1417]: Succeeded, trace file is /tmp/pam_test-trace.1417
```

Chapter 8: User Synchronization

This section contains the following topics:

[User Synchronization between Servers](#) (see page 99)

[Endpoint Account to Global User Synchronization](#) (see page 102)

[Synchronize Users in Create or Modify User Tasks](#) (see page 107)

[Account Synchronization](#) (see page 108)

User Synchronization between Servers

You configure synchronization in Identity Manager to make sure that the users for corporate directory and provisioning directory have matching data. To handle changes from either directory, you configure inbound and outbound synchronization.

Inbound Synchronization

Inbound synchronization keeps Identity Manager users up to date with changes that occur in the provisioning directory. Changes in the provisioning directory include those made using Provisioning Manager or systems with connectors to the Provisioning Server. The synchronization uses the mappings defined on the Provisioning screen of the Management Console.

Outbound Synchronization

Outbound synchronization involves using Identity Manager to create and update users in the provisioning directory.

Create Global Users from Identity Manager

User creation in the provisioning directory occurs only for provisioning related events, such as assigning a provisioning role to a user. No user is created in the provisioning directory when you use an admin task to create a user unless that task assigns a role or includes an identity policy that assigns the role.

When user creation in Identity Manager triggers user creation in the provisioning directory, Identity Manager sends an email with a temporary password to the new user's email address as it is defined in the provisioning directory. The user can log into the User Console with that password, however, the user is then required to change to a new password. As a result, the password is synchronized between the user store and provisioning directory.

If the user has no email address, the user cannot access the User Console until changing password in the user store, or an Identity Manager administrator changes the user's password in the Provisioning Manager.

Note: To email a temporary password, email notifications must be enabled for the Environment, and the `CreateProvisioningUserNotificationEvent` must be configured for email notification. (See the *Configuration Guide*.)

Update Global Users using Identity Manager

Updates to users in the provisioning directory occur when you use an admin task that modifies users. If no global user exists, no synchronization occurs.

Outbound mappings match the Identity Manager user events to an outbound event that affects the provisioning directory.

Identity Manager User Event	Outbound Event
☐ DeleteUserEvent	POST_DELETE_GLOBAL_USER
☐ DisableUserEvent	POST_DISABLE_GLOBAL_USER
☐ EnableUserEvent	POST_ENABLE_GLOBAL_USER
☐ ModifyUserEvent	POST_MODIFY_GLOBAL_USER
☐ ResetPasswordEvent	POST_CHANGE_GLOBAL_USER_PWD

If a user exists in the provisioning directory but not in Identity Manager, you can create that user in the User Console. If you have mapped attributes for the create task and the users have the same user ID, the attributes for the provisioning user are updated in the provisioning directory. Now you can manage that user from Identity Manager.

Note: If an event updates user attributes and you want the values to be synchronized to Identity Manager, then you need to map the events to the Outbound Event: `POST_MODIFY_GLOBAL_USER`.

Delete Global Users using Identity Manager

By default, outbound synchronization is configured for the Delete User event. When you delete a user in Identity Manager, the user is also deleted in the provisioning directory and all endpoint accounts.

If Identity Manager cannot delete a user's account in a managed endpoint, it deletes the user from the remaining accounts, but does not delete the user from the provisioning directory.

For example, suppose User A has a UNIX account and an Exchange account, which are managed in the Provisioning Server. When user A is deleted in Identity Manager, the Provisioning Server attempts to delete the user's accounts. If the Provisioning Server cannot delete the Exchange account due to a communication error, it deletes user A's UNIX account, but does not delete the user from the provisioning directory. However, User A is not restored in the user store.

Enable Password Synchronization

The Provisioning Server allows password synchronization between Identity Manager users and associated endpoint user accounts. Two configurations are required to enable endpoint initiated changes:

- Endpoints must be configured to capture endpoint-initiated changes and forward the changes to the Provisioning Server.
Note: For more information on how to configure endpoints for password synchronization, see the *CA Identity Manager Administration Guide*.
- The Enable Password Synchronization Agent attribute should be activated for the Global User.

To enable password synchronization

1. In the Management Console, choose Advanced Settings, Provisioning.
2. Check Enable Password Changes from Endpoint Accounts.
3. Click Save.
4. Restart the Application Server.

More Information

[Password Synchronization](#) (see page 75)

Endpoint Account to Global User Synchronization

The Provisioning Server performs the following types of synchronization:

User Synchronization

Ensures that each global user has the necessary accounts on the appropriate managed endpoints, and that each account is assigned to the appropriate account templates as called out by the global user's provisioning roles.

Account Synchronization

Ensures that the capability attribute values on accounts are the appropriate values as indicated by the account's assigned account templates. Account synchronization can be strong or weak. Weak synchronization ensures that accounts capability attributes have at least the minimum capability required by its account templates. Strong synchronization ensures that account capability attributes have the exact capability required by its account templates. Account synchronization is strong if the account belongs to at least one account template whose Strong Synchronization check box is selected.

No corresponding Strong Synchronization check box governs User Synchronization, but a similar concept exists. When you issue the Synchronize User with Roles menu item on a global user, you are presented with two synchronization options:

- Add missing accounts and account template assignments.
- Delete extra accounts and account template assignments.
- By selecting only the Add check box, which is similar to Weak Account Synchronization, you want global users to have at a minimum all accounts required by their assigned provisioning roles, but you allow global users to have additional accounts not prescribed by current provisioning roles.

Select both the Add and Delete check boxes, which is similar to Strong Account Synchronization, to have the provisioning roles define exactly which accounts the global user should have. Any additional accounts are deleted.

Choose Weak/Strong Account Synchronization or Weak/Strong User Synchronization based on how precisely provisioning roles are defined. If your users fit into clearly-defined provisioning roles where account access is tied to those roles, you would use Strong Synchronization.

Note: Some endpoint types set strong synchronization as the default. For more information, see your endpoint type-specific *Connector Guide*.

User synchronization and account synchronization are separate tasks that you must perform individually. Typically, you perform user synchronization first to ensure that all necessary accounts are created, then perform account synchronization later so the Provisioning Server assigns or changes the values of the account attributes.

The Provisioning Server provides two sets of synchronization menu options for objects:

- Check synchronization menu options verify the synchronization and return a list of the accounts that do not comply with the provisioning roles or account templates.
- Synchronize menu options synchronize global users with their provisioning roles or accounts with their account templates.

If you perform the check synchronization functions first, the Provisioning Server tells you what corrections the synchronize functions will perform. If the check synchronization functions find no problem, the synchronize functions do not run.

Note: You need not run the check synchronization functions first. However, because you are becoming familiar with the Provisioning Server, we recommend that you run these commands so you learn what to expect from synchronization functions.

Why Global Users Become Out of Sync

The following are some reasons why global users become out of sync with their provisioning roles or account templates:

- Earlier attempts to create the necessary accounts failed due to hardware or software problems in your network, thereby causing missing accounts.
- Provisioning roles and account templates may have changed, thereby creating extra or missing accounts.
- Accounts were assigned to account templates after they were created, so accounts exist that have not been synchronized with their account templates.
- The creation of a new account is delayed because the account was specified to be created later.
- A new endpoint was acquired. During exploration and correlation, the Provisioning Server does not assign provisioning roles to the global users automatically, so you must update the role to indicate which users should have accounts on the new endpoint. Any account that was correlated to a global user is listed as an extra account when the global user is synchronized.

- An existing account was assigned to a global user by copying the account to the global user, thereby performing a manual correlation and establishing an extra account.
- An account was created for a global user other than by assigning the user to a role. For example, if you copy a global user to an account template that is not in any of the user's provisioning roles, the account is listed as an extra account or as an account with an extra account template. If you copy the global user to an endpoint to create an account using the endpoint's default account template, that account could be an extra account.

Global User Synchronization

User synchronization creates, updates, or deletes accounts so they comply with the provisioning role assigned to the global user. Internally, User Synchronization runs a Check User Synchronization operation to identify accounts that need to be created, deleted or updated. Then it carries out those identified actions. Check User Synchronization does its work solely in information found in the provisioning directory. This is where the Provisioning Server records which accounts exist, which accounts are associated with which global users and which account templates are assigned to which accounts. If your administrators add or delete accounts on your managed endpoint by using native tools, and you have not performed a recent re-exploration of your endpoint to update the provisioning directory, User Synchronization may indicate there are no problems when a user may have extra or missing accounts. To address missing accounts, see the section Recreating Deleted Accounts.

User synchronization does not perform account synchronization unless there are problems reported by Check User Synchronization. After a successful user synchronization, you may need to run account synchronization to ensure your account attributes are in synch with the account's account templates.

Create Accounts

Because provisioning roles contain account templates, and account templates are associated to endpoints, a global user should have accounts listed on each endpoint with the correct account attributes.

During the user synchronization process, if Identity Manager discovers that a required account does not exist on an endpoint, it creates the account on the endpoint. If more than one account template in the global user's provisioning roles prescribes the same account, the account is created by merging all relevant account templates. This account is assigned to those account templates, which are currently not synchronized with the account. Account synchronization is not necessary on newly created accounts.

Add Account Templates to Accounts

If an existing account is missing one or more account template assignments, user synchronization assigns an existing account to those Account Templates. When an account is assigned to one or more new Account Templates, account synchronization is run automatically to update the capability attributes of the account to capabilities specified by the Account Templates.

After account update from user synchronization, the account may or may not be in sync with its Account Templates. If one of the Account Templates added was a strong synchronization account template or if two or more Account Templates were added to an account, user synchronization will start a full account synchronization on the account. However, if only one weak synchronization account template was added, user synchronization starts an account synchronization involving only this one account template. If the account was previously out of account synchronization with its other Account Templates before this update, it could still be out of account synchronization afterwards.

Delete Accounts

During user synchronization, you have the option to delete extra accounts. You may determine that your users have legitimate reasons for having accounts other than those required by their provisioning roles. If that is the case, you should not select this delete option.

If an account being deleted resides in a managed endpoint for which account deletions have been disabled, the account is not actually deleted. See the section Using Delete Pending.

Removing Account Templates from Accounts

User synchronization can also be used to remove extra account templates from an account. This is only done if you select the delete option. When user synchronization determines that an account needs to be updated to remove one or more extra account templates, account synchronization is run automatically on the account to synchronize its capability attributes with the account templates remaining on the account.

This account synchronization that occurs when removing account templates from an account will use strong synchronization if any of the remaining account templates is marked for strong synchronization and weak synchronization if all of the remaining account templates are marked for weak synchronization.

Whether weak or strong synchronization is used affects whether account capabilities granted earlier when an account template was assigned to an account are taken away when that account template is later removed. With strong synchronization, a capability granted by an account template, such as a group membership or higher quota, will be taken away (group membership removed or quota lowered) if none of the account templates remaining on the account prescribe that capability. However, with weak synchronization, typically the account is unchanged because the Provisioning Server does not distinguish between on-demand extra capabilities and capabilities granted through account templates.

The exception to this rule is for certain multivalued capability attributes designated as SyncRemoveValues attributes. A simple multivalued attribute representing a collection of values assigned to the account (a group membership list, say), will typically be listed as a SyncRemoveValues attribute. For these attributes, the weak synchronization action that occurs while removing an account template from an account will remove values prescribed by the account template that is being removed - as long as that value is not also prescribed by one of the remaining account templates.

For example, if you create your account templates where each account template assigns a unique group membership to your account, this SyncRemoveValues feature will mean that when you change a global user's provisioning roles so as to no longer require a particular account template, the account will be updated to no longer belong to the group prescribed by that account template. You will note that this is not exactly the same as strong synchronization, as group memberships given to accounts beyond what is prescribed to account templates are retained.

For all single-valued attributes and certain multivalued attributes which are not designated as SyncRemoveValues attributes, the weak synchronization action while removing an account template from an account is the same as a normal weak synchronization action - capabilities are never removed.

See the file eTaCapability.txt file for an endpoint type-by-endpoint type list of which account attributes are capability attributes and which of these capability attributes have been designated as SyncRemoveValues attributes.

To create the eTaCapability.txt file run the following command:

```
..\Provisioning Server\bin\dumpptt.exe -c > eTaCapability.txt
```

If you want the capabilities never to be removed by weak synchronization, disable the SyncRemoveValues feature by setting the domain configuration parameter Synchronize/Remove Account Template Values from Accounts to No.

Check User Synchronization

You can check synchronization on global users or provisioning roles to list accounts that are missing. For global users, the list may include extra accounts or account templates. You can select synchronization checking for a global user or a provisioning role.

The following table describes what the Provisioning Server looks for when you check user synchronization:

Object	Action
Global user	Ensures that the global user has all the accounts required by the person's provisioning roles and ensures each account belongs to the correct account templates.
Provisioning Role	Ensures that each global user assigned to the provisioning role has all the accounts required. It also ensures that each account belongs to the account templates associated with the role.

Synchronize Global Users or Roles

You can perform user synchronization on global users or roles. Synchronization is a choice on the right-click pop-up menu for a global user or a role. The following table describes what the Provisioning Manager does when it synchronizes global users or roles:

Object	Action
Global user	Synchronizes the global user with each role associated to it.
Role	Synchronizes the role with each global user associated to it.

Synchronize Users in Create or Modify User Tasks

On the profile tab of a task that creates or modifies users, synchronization controls ensure that changes to the Identity Manager are also made to the global user. If you create admin tasks that create or modify users and you have Identity Policies, set the synchronization controls as follows:

- Set User Synchronization to On Task Completion.
- Set Account Synchronization to On Task Completion.

Note: For best performance, select the On Task Completion option. However, if you select the On Task Completion option for a task that includes multiple events, Identity Manager does not synchronize until all of the events in the task complete. If one or more of those events require workflow approval, this may take several days. To prevent Identity Manager from waiting to apply identity policies or synchronize accounts until all events complete, select the On Every Event option.

If you add attributes to admin tasks that manage users, you need to update the Attribute Mappings in the Provisioning screen in the Management Console. For each user attribute in Identity Manager, a default provisioning attribute exists.

User Attribute	Provisioning Attribute
☐ %ADMIN_ROLE_CONSTRAINT%	%ADMIN_ROLE_CONSTRAINT%
☐ %EMAIL%	%EMAIL%
☐ %ENABLED_STATE%	%ENABLED_STATE%
☐ %FIRST_NAME%	%FIRST_NAME%
☐ %FULL_NAME%	%FULL_NAME%
☐ %IDENTITY_POLICY%	%IDENTITY_POLICY%
☐ %LAST_NAME%	%LAST_NAME%
☐ %PASSWORD%	%PASSWORD%
☐ %PASSWORD_DATA%	%PASSWORD_DATA%
☐ %USER_ID%	%USER_ID%

Account Synchronization

Account synchronization updates capability attributes to ensure that the account has the capabilities specified by the account templates.

Note: Synchronization updates an account's capability attributes, but does not affect the account's initial attributes.

Initial attributes and capability attributes are updated by the Account Template Attribute Changes dialog. When you update capability attributes in an account template, and then apply the changes to the accounts, the attributes are updated automatically.

To synchronize capability attribute changes in an account template with its accounts, use the Account Template Attribute Changes dialog or one of the synchronization menu options discussed in this section.

Check Account Synchronization

You can check account synchronization for accounts, containers, endpoints, global users, account templates, and roles. This action returns a list of accounts that do not comply with account templates. The following table describes what happens when you check the synchronization of accounts on each object:

Object	Synchronizes
Account	Account attributes and ensures they comply with associated account templates.
Container	Account attributes for each account in the container and ensures they comply with associated account templates.
Endpoint	Account attributes for each account on an endpoint and ensures they comply with associated account templates.
Global user	Account attributes for each of a global user's accounts and ensures they comply with associated account templates.
Account Template	Account attributes for each account associated with this account template and ensures they comply with the account template. If the account template uses strong synchronization, the account attributes for each account are checked so they comply with all associated account templates.
Role	Account attributes for each account associated with an account template included in this role and ensures they comply with the account templates in the role.

Synchronize Accounts

You can perform account synchronization on accounts, containers, endpoints, global users, account templates, and roles. The following table lists the effect of account synchronization on each object:

Object	Synchronizes
Account	The account with its associated account templates.
Container	Each account in a container with its associated account templates.
Endpoint	Each account on an endpoint with its associated account templates.
Global user	Each account of a global user with each account template associated to it.

Object	Synchronizes
Account Template	Each account associated with the account template.
Role	Each account with each account template in a role.

Chapter 9: etautil Batch Utility

You use the etautil batch utility to perform the same tasks as you do with the Provisioning Manager, but from a command line interface. This utility is especially useful for performing repetitive and time-consuming tasks. This chapter explains etautil and provides examples of its use.

Note: etautil sometimes uses the original terminology associated with eTrust Admin, such as namespace and policy, instead of endpoint type and account template. This occurs when you use actual LDAP schema items (object class names, attribute names, attribute values) which retain the original terminology for backwards compatibility. However, etautil also allows the use of user-friendly attribute names as specified in parser tables. These names use the new Identity Manager terminology.

This section contains the following topics:

[Tasks You Can Perform](#) (see page 111)

[etautil Syntax](#) (see page 112)

[Use DeletePending](#) (see page 120)

[Common Error Messages](#) (see page 121)

[Obtain Operation Details](#) (see page 122)

[DOS Output from etautil](#) (see page 123)

Tasks You Can Perform

You can use etautil to maintain property sheets and inclusion pages for Provisioning Manager objects. The following are tasks you can perform with the Batch Utility:

- Create a batch file to explore and correlate endpoint accounts
- Synchronize several accounts with the account template to which they are assigned
- Search and replace attribute values for a large set of objects

For more information about the rules for control statements to use with etautil, see the Provisioning Manager help. For details on using etautil with a specific connector, see the *Connectors Guide*.

etautil Syntax

This is the generalized syntax of etautil. For an explanation of the syntax and use of etautil, see the Provisioning Manager help.

```
etautil [-n] [-d domain] [-u user [-p password]] [-y password-file] [options]  
control_statements
```

Note: Using an input file provides better performance. A single bind executes all commands in the file.

-n

Verifies the syntax of the command you entered, without executing the command.

-d domain

Specifies the name of the provisioning domain.

-u user

Specifies the global user name for authentication.

-p password

Specifies the password of the named global user for authentication.

Cannot be specified with `-y password_file` option

-y password-file

Specifies a file name that contains a global user password. Cannot be specified with `-p password` option. Please see the "Important" section below for more information.

options

Includes any of the following:

-f filename

Reads the control statements in the indicated file and executes them. Use semicolons (;) to delimit multiple control statements.

-i

Invokes the etautil interactive mode, which lets you enter control statements at the prompt. (Use <Ctrl+D> or <Enter> to terminate the interactive mode).

-o

Displays operation details to stdout. See the section Obtain Operation Details.

-h

Displays etautil help.

control statements

For more information about control statements, see etautil Control Statements.

Important: Enter all DNs in the same case as stored in the provisioning directory. DNs are strings that etautil often requires in your commands. In most cases, an incorrect-case DN supplied to the Provisioning Server is processed as is. Authorization errors are common as most permission checking is done by a case-sensitive comparison of the DN of an object being operated upon with the DN specified in a privilege. Copying DN strings from logs or the JXplorer utility ensures the DN is in the correct case.

On UNIX, we strongly recommended you include the -y password-file option to specify an authentication password. For example, if "\$HOME/.pwdfile" contains myglobaluser's password, then you can use etautil command as follows:

```
$ etautil -u myglobaluser -y "$HOME/.pwdfile" <other-options>
```

The command disregards any newline character if one exists at the end of the password file, but it uses the rest of the content as the authentication password.

etautil Control Statements

Control statements tell etautil the procedures to carry out; this is the request that is sent to the Provisioning Server. Use semicolons to delimit multiple control statements in a single etautil command.

Each statement must begin with a verb followed by a base distinguished name (base dn), an object's class name, and the object's operands.

```
verb basedn classname operands
```

Note: For more information about control statements see the Provisioning Manager help. For endpoint type-specific details, see the *Connector Guide* for your endpoint type.

The following are examples of the etautil control statements:

ADD

The following example creates role-based accounts for a user:

```
add 'eTGlobalUserContainerName=Global Users,eTNamespaceName=CommonObjects'
    eTGlobalUser globalusername=denro01 in
    'eTRoleContainerName=Roles,eTNamespaceName=CommonObjects' eTRole
    RoleName=TeamManager
```

The following example register a UNIX endpoint:

```
add 'eTNamespaceName=UNIX - etc' eTETCDirectory name=hpdevsrv
    eTETCHost=hpdevsrv eTETCUnicenterSec=0 eTETCUnicenterUser=0
```

The following example creates a global user named HAAS14 and assigns the values of myvalue1 and myvalue2 to the custom fields with the IDs of 01 and 02.

```
etautil -u etaadmin -p super**s add 'eTGlobalUserContainerName=Global
    Users,eTNamespaceName=CommonObjects'
    GlobalUserName=user14 eTCustomField01=myvalue1 eTCustomField02=myvalue2
    eTPassword=super**s eTUserId=user14
```

Note: You cannot use the ADD statement to add mainframe endpoints to the Provisioning Server.

The following example creates an inclusion between an LDAP account template defined in the domain ETACHILD and a role defined in the domain ETAROOT:

If the command is launched on the ETAROOT server:

```
add 'eLDAPPolicyContainerName=LDAP
Policies,eTNamespaceName=CommonObjects@ETACHILD' eLDAPPolicy name=LDAPol in
'eTRoleContainerName=Roles,eTNamespaceName=CommonObjects' eTRole
RoleName=Role
```

If the command is launched on the ETACHILD server:

```
add 'eLDAPPolicyContainerName=LDAP Policies,eTNamespaceName=CommonObjects'
eLDAPPolicy name=LDAPol in
'eTRoleContainerName=Roles,eTNamespaceName=CommonObjects@ETAROOT' eTRole
RoleName=Roles
```

In all cases:

```
add 'eLDAPPolicyContainerName=LDAP Policies,eTNamespaceName=CommonObjects'
eLDAPPolicy name=LDAPol in
'eTRoleContainerName=Roles,eTNamespaceName=CommonObjects@ETAROOT' eTRole
RoleName=Roles
```

COPY/COPYALL

Copy creates a new global user with the same properties as an existing global user, including the same roles.

Copyall performs the same function as Copy but also copies the existing user's relationships (inclusions) to the new global user.

Syntax:

```
copy|copyall 'eTGlobalUserContainerName=Global
Users,eTNamespaceName=CommonObjects' eTGlobalUser
globalusername=existing_user[.domain] to
globalusername=new_user eTFullName='new fullname'
[property1=value property2=value ... propertyn=value]
```

Example:

```
copyall 'eTGlobalUserContainerName=Global Users,eTNamespaceName=CommonObjects'
eTGlobalUser globalusername=user01 to globalusername=user12 FullName='John
Doe'
Password=password EmailAddress=JohnDoe@mycompany.com
```

DELETE

Deletes a global user and its relationships from an endpoint. To delete an object and its inclusion objects, the syntax is:

```
delete basedn classname namingattribute=value
```

To delete an inclusion object, the syntax is:

```
delete childbasedn childclass childnamingattribute=value in parentbasedn  
parentclass parentnamingattribute=value [relationship=rel]
```

Note: The deletion of a global user and its accounts can be done using the Update control statement described later in this chapter.

EXPLORE

Finds objects in a registered endpoint and stores them in the provisioning directory. Optionally, correlates or creates a global user in the Provisioning Server for the person associated with each account in the endpoint.

Syntax:

To explore an entire endpoint, the syntax is:

```
explore dirbasedn dirclassname dirnamingattribute=value list [explore  
options]
```

To explore only a specific container, the syntax is:

```
explore base_dn_container_class_name name=container_name [scope=value] list  
explore_options
```

The *explore_options* include the following:

- **ExploreUpdateEtrust**-Retrieves all managed objects.
- **ExploreCorrelateUsers**-Correlates accounts with global users using existing ones.
- **ExploreCreateUsers**-Creates global users as needed during the correlation. **ExploreUpdateUsers**-Sets/refreshes the global user attributes using account attribute values.

Note: Combining explore, correlate, and update actions into a single request is not supported.

Examples:

To explore and correlate an entire UNIX endpoint using existing global users:

```
explore 'eTNamespaceName=UNIX - etc' eTETCDirectory
name= hpserv01 list eTExploreUpdateEtrust

explore 'eTNamespaceName=UNIX - etc' eTETCDirectory
name= hpserv01 list eTExploreCorrelateUsers
```

To explore a specific NDS container:

```
explore 'eTNDSTreeName=SampleTree,eTNamespaceName=NDS
Servers'
eTNDSTreeName=SampleTree scope=1
list ExploreUpdateEtrust
```

MASSCHANGE

Sets the same attribute values on a set of objects or searches and replaces attribute values on a set of objects.

Syntax:

```
masschange basedn class criteria [scope=value] to property0=value
[property1=value... propertyn=value]
```

where:

criteria-Is the filter for selected target objects.

Scope-Specifies the scope of the search operation (1 for 1-level, 2 for sub-tree level; the default is 1).

propertyn=value-Specifies the attribute to be updated and its new value.

Example:

This example replaces the string (310) with (424) in the eTTelephone value and sets the eTStreetAddress to 15 Software Street for the global users who have eTCity equal to Santa Monica and a name beginning with u:

```
masschange 'eTGlobalUserContainerName=Global Users,
eTNamespaceName=CommonObjects' eTGlobalUser City='Santa Monica'
GlobalUserName=u* to Telephone=#sp(310)p(424) StreetAddress='15 Software Street'
```

REPORT

Use REPORT to check account or user synchronization. For more information, see Report Accounts that Do Not Comply with Account Templates.

Syntax:

```
report basedn class namingattr=value list reporting_attribute
```

reporting_attribute-Must be eTSyncAccounts, eTSyncUsers, or eTSyncDelete.

Example:

This example reports all existing accounts that do not comply with the account templates to which they are assigned for the global user ayrton02:

```
report 'eTGlobalUserContainerName=Global Users,  
eTNamespaceName=CommonObjects' eTGlobalUser globalusername= user02 list  
eTSyncAccounts
```

UPDATE

Use the Update control statement to do the following:

- Synchronize accounts with account templates.
- Suspend and resume a global user. You can specify that all accounts associated with the global user are also suspended or resumed.
- Change the attributes of an account template and apply those changes to the associated accounts. To propagate those changes to each account assigned to the account template, specify the phrase eTSyncAccounts=1.
- Delete a global user, its relationships, and its accounts.
- Update the attributes values of an existing object.

Syntax:

```
update basedn class namingattribute=value to entries
```

Examples:

To synchronize an account synchronization for a role:

```
update 'eTRoleContainerName=Roles,  
eTNamespaceName=CommonObjects'  
eTRole RoleName=F1Drivers to eTSyncAccounts=1
```

To delete a global user and its accounts:

```
update 'eTGlobalUserContainerName=Global  
Users,eTNamespaceName=CommonObjects'  
eTGlobalUser globalusername=user02 to DeleteUserAndAccounts=1
```

To remove a value of a multivalued attribute such as eTRoleDN:

```
update 'eTGlobalUserContainerName=GlobalUsers,eTNamespaceName=CommonObjects'  
GlobalUser GlobalUserName=y272705 to -eTRoleDN=  
'eTRoleName=LNDSuspended,eTRoleContainerName=Roles,eTNamespaceName=  
CommonObjects,yourdomainsuffix'
```

This command example uses a plus (+) or minus (-) sign operator in the update section to add or remove values of a multivalued attribute. In this example, there is a minus sign (-) operator before the eTRoleDN attribute to delete an association between Global User and Role.

Multivalued Attributes

Each provisioning role, account template, and global user is an object. Each object has attributes, some of which are multivalued. For example, a Global User may belong to multiple Roles. You may need to update or delete the values for these attributes using plus sign (+) or minus sign (-) operators with the UPDATE command using the following syntax:

```
+attribute_name=attribute_value for adding a value  
-attribute_name=attribute_value for removing a value  
attribute_name=attribute_value for replacing existing value(s) by a new one  
attribute_name='' for clearing existing value(s)
```

Multivalued attributes include the following:

- Account Objects
 - eTPolicyDN (the list of account templates assigned to an account)
 - endpoint type-specific group membership attributes
- Account Template Objects
 - endpoint type-specific group membership attributes
- Global Users Objects
 - eTUserAdminProfile (the list of assigned admin profiles)
 - eTCustomField01 through eTCustomField99 (all global user custom attributes are multivalued)
 - eTAccessControlList (the list of privileges the global user has)

Note: For more information about multivalued attributes, see the *Connectors Guide*.

Use DeletePending

To designate an account as DeletePending, you set two endpoint attributes:

eTAccountDeletable

controls what action the Provisioning Server performs when accounts on an endpoint are deleted. The values are:

0--Enable DeletePending to suspend an account and mark it for later deletion.

1--Disable DeletePending and physically delete the account from the managed endpoint. This is the default value.

2--Enable an alternate delete behavior to remove an account from the Provisioning Server but leave the account unchanged on the managed endpoint.

eTAccountForcedDeletable

controls whether or not an account marked for DeletePending can be deleted through the Forced Delete operation. Use these values:

0--Disable ForcedDelete on DeletePending accounts. This is the default.

1--Enable ForcedDelete on DeletePending accounts.

To track accounts that have been suspended or are in a DeletePending state, use these attributes:

- eTSuspendedDate is the date the account was suspended using the Provisioning Server.
- eTSuspendedTime is the time the account was suspended using the Provisioning Server.
- eTSuspendedReason is either DeletePending or AdminSuspended.

Note: These attributes are only set when an account is suspended using the Provisioning Server. If an account is acted upon by the native endpoint type tools, these attribute values will be stale. If you are taking action based on these attributes, use eTSuspended to confirm whether an account is actually suspended.

Common Error Messages

The following are some common error messages associated with etautil:

Unknown error nnn opening Common Object Repository

This message appears when the authentication to the Provisioning Server fails. If the nnn value in the message is:

102, the user/password is wrong (-u/-p).

96, the domain is wrong (-d).

End of file reached while expecting an operator

Etautil cannot parse the control statements correctly. This message appears when the grammatical syntax is not respected.

Object 'XXXX' operation failed: DB operation failed: Target DN not found.

The target object cannot be reached. This message appears when the base dn is not correct (wrong value for the components of the dn).

Object 'XXXX' operation failed: No server plug-in found for operation

The Provisioning Server is not able to find the connector server corresponding to XXXX. This message appears when XXXX is not correct.

Class 'classname' is not a valid class name

The LDAP name or user-friendly name class name is not defined in the corresponding endpoint type.

Could not find keyword xxxxx for class classname

The LDAP name or user-friendly name xxxxx does NOT correspond to an attribute defined in the class classname. The problem is on 'xxxxx', not on classname.

Obtain Operation Details

You can use the following methods to obtain operation details:

- Specify the -o argument with etautil to display operational details to your standard output device (stdout).
- Specify the OpDetail attribute to control operation detail on a command-by-command basis. By setting this attribute to 0 or 1 you can determine the commands for which you receive operation details. For SELECT and EXPLORE commands, you must set OpDetail in the filter, for example:

```
explore 'eTNamespaceName=Windows NT' eTN16Directory name='My NT Directory'
OpDetail=1 List eTExploreUpdateEtrust
```

For other commands such as UPDATE, you must set OpDetail in the attribute list, for example:

```
update 'eTGlobalUserContainerName=Global Users,eTNamespaceName=CommonObjects'
eTGlobalUser GlobalUserName=gluser01 to
LastName='gluser01 lastname' OpDetail=1 eTSyncAccounts=1
```

Note: You can combine both methods to obtain operation details for only some of the commands defined in an input file.

This example makes use of a batch input file to run multiple commands that explore a Windows NT endpoint and update a global user name:

```
etautil -o -u etaadmin -p password -f myinputfile
```

where myinputfile contains the following syntax:

```
explore 'eTNamespaceName=Windows NT' eTN16Directory name=My NT Directory  
OpDetail=0 List eTExploreUpdateEtrust;  
update 'eTGlobalUserContainerName=Global Users,eTNamespaceName=CommonObjects'  
eTGlobalUser GlobalUserName=gluser01 to LastName='gluser01 lastname'  
eTSyncAccounts=1
```

Note: This example makes use of the -o flag to display operation details to stdout. To control the amount of information displayed, use the OpDetail attribute. In this example, by setting OpDetail=0 in the Explore command, only the details of the Update command are displayed.

DOS Output from etautil

When commands are issued directly from the DOS command prompt, non-ASCII 7-bit (ENU) characters are not converted correctly in etautil. This problem occurs because the character set used by DOS (EOM) and Windows (ANSI) are different. The following is a workaround:

- For single-byte non-ASCII (ENU) characters, redirect the output of the etautil command to a text file.
- For multi-byte non-ASCII (ENU) characters, use an input file that contains etautil control statements you want to execute. Also, redirect the output to a text file.

```
etautil [-d DomainName] -u UserName -p Password control statement >  
Output.txt
```

```
etautil [-d DomainName] -u UserName -p Password -f Input.txt > Output.txt
```


Chapter 10: SPML Service

This section contains the following topics:

[SPML Overview](#) (see page 125)

[Install SPML](#) (see page 132)

[Uninstall the SPML Service](#) (see page 132)

[SPML Service Configuration](#) (see page 132)

SPML Overview

The Provisioning Server helps you manage, provision, and de-provision entities. A good provisioning system is vital for security and efficiency. Many companies have multiple provisioning systems. It can often be difficult to configure different provisioning systems to communicate with each other.

OASIS (Organization for the Advancement of Structured Information Standards) has developed a markup language specifically designed to facilitate communications between and within user provisioning software. This is named SPML (Service Provisioning Markup Language).

SPML is an open standard that provides an XML-based protocol for provisioning requests. It facilitates provisioning requests between clients and servers that can be both intranet and extranet.

Benefits of Using SPML

The benefits of the SPML include the following:

- SPML is an open standard and can therefore communicate with other provisioning systems that can process SPML Requests. This lets businesses continue to use and integrate existing systems.
- Data can be shared across different provisioning systems to leverage the best features of each system.
- SPML is especially designed to handle provisioning-related data.
- SPML can easily handle data driven assignments of role-based access control.
- SPML is a best-of-breed technology for user provisioning.
- SPML facilitates business-to-business communications, where appropriate.

- SPML XML requests and responses are more human-readable than LDAP. Requests which is the native language of the Provisioning Server.
- SPML is a web-based technology.

When You Would Use the SPML Service

SPML simplifies provisioning requests and facilitates communication between provisioning systems. You do not need to deploy the SPML Service for a basic Provisioning Server installation, but it is an efficient and elegant provisioning solution.

You would deploy the SPML Service and the associated SPML Configuration Application as part of an installation or upgrade of the Provisioning Server. You can then deploy the SPML clients and tools that come with the Provisioning Server (CMDRA, SPML Manager, and WS-Mapper). You can also write or integrate a third-party clients and requesting authorities if they support the SPML version 1.0 standard.

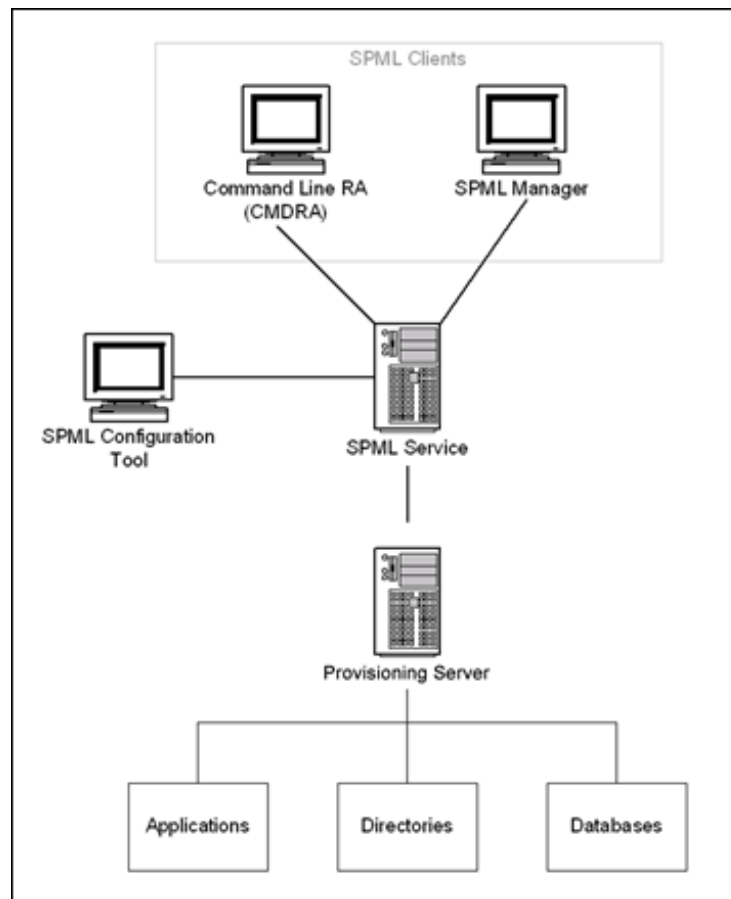
SPML Architecture

The SPML Service is the server-side component that processes SPML requests. The SPML Service is a Provisioning Server component that uses and processes SPML requests. The SPML Service uses SPML version 1.0.

This section describes the components that make up the SPML architecture of the Provisioning Server.

SPML Architectural Diagram

The following diagram shows the SPML components of the Provisioning Server and how they relate to each other.



SPML Service

The SPML Service is the server-side component that processes SPML requests. The SPML Service is a Provisioning Server component that uses and processes SPML requests. The SPML Service uses SPML version 1.0.

SPML Configuration Application

The SPML Configuration Application is a web-based interface that lets you configure one or more Provisioning Servers as unique instances of SPML services. You should use the SPML Service Configuration tool to configure the SPML Service.

The SPML Configuration Application is automatically installed when you install the SPML service.

To access the SPML Configuration Application, Start Menu, Programs, CA, Identity Manager, IM SPML Service Configuration.

Command Line Requesting Authority (CMDRA)

The Command Line Requesting Authority (CMDRA) is a sample SPML Requesting Authority that can submit well-formed SPML Request XML files to the SPML Web Service. The CMDRA lets advanced users submit SPML requests using the command line or from scripts. It is ideal for sorting and managing large amounts of data using SPML templating, as well as automating requests and large batch jobs.

To download the CMDRA

1. Click Start Menu, Programs, CA, Identity Manager, IM SPML Requesting Authority
2. Click the cmdra.zip link.
3. Unzip the CMDRA to your hard disk to use it.

SPML Manager

The SPML Manager is a graphical user interface that lets administrators create and execute SPML provisioning requests. The SPML Manager can also help advanced users integrate with other provisioning systems. You can design provisioning requests in the SPML Manager then view the SPML requests in its native XML format.

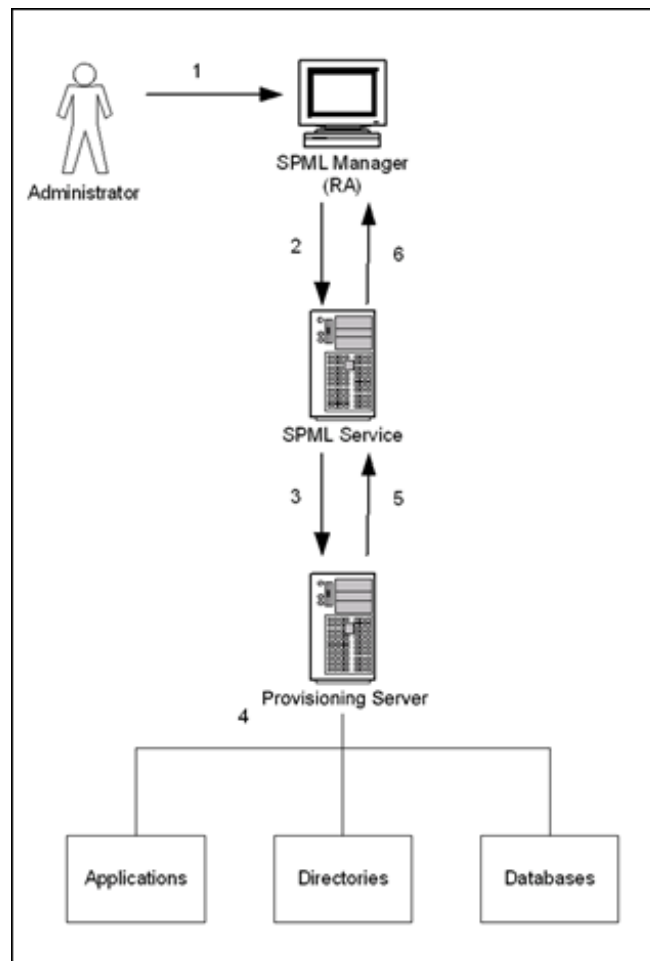
To download the SPML Manager

1. Click Start Menu, Programs, CA, Identity Manager, IM SPML Requesting Authority
2. Click the SPML Manager web link.
3. Unzip the SPML Manager to your hard disk to use it.

How the SPML Service Works

This section explains how an SPML request works from the Requesting Authority through to the provisioning server.

1. Using the SPML Manager, an administrator creates a new user.
2. The SPML Manager creates an SPML-specific XML form and sends it to the SPML service.
3. The SPML Service translates the request to LDAP and passes the LDAP request to the Provisioning Server.
4. The Provisioning Server processes the LDAP request.
5. The Provisioning Server sends confirmation to the SPML Service.
6. The SPML Service sends confirmation to the SPML Manager.



SPML Integration

This section gives you an outline of the following:

- SPML Templating to integrate large amounts of data from external provisioning systems
- The WS Mapper tool to convert protocols
- Requesting Authorities to connect to the Provisioning Server through the SPML Service

SPML Templates

SPML technology lets you create templates that can be applied to data that is being imported into the Provisioning Server through the SPML Service. You can also use SPML Templating to modify the data as it goes into the Provisioning Server.

The SPML template takes records from a CSV or XML files and applies the template to each record. Because the template is applied to each record, you can impose some rule-based conditions to each record that affects the data output. The SPML output is then fed to the SPML Service and then sent to the Provisioning Server.

The templates are written using Velocity. Velocity is an open source Java-based engine that is specifically designed for processing templates. You can create your own templates or use the sample template that comes with the SPML Manager or the CMDRA.

You can write templates using the SPML Manager, or you can code them yourself. You can import mass data into the Provisioning Server using the templating functionality with the SPML Manager, SPML Feed, or the CMDRA.

The SPML service comes with example templates and example data files to help you understand and create your own templates.

For more information on SPML Templates, see using the SPML Manager's Templating Functionality.

WS Mapper

WS Mapper (Web Service Mapper) is a lightweight web service that takes a proprietary web service request and transforms the data into another web service request format. The service can also transform the response to the web service request back into the original format.

This service was designed to allow web service requests from third-party applications to be redirected to this service and mapped into SPML Service provisioning requests that will end up as provisioning tasks in the Provisioning Server.

Requesting Authorities

Any Requesting Authority client that uses standard SPML 1.0 can send provisioning requests to the Identity Manager SPML Service. The SPML Service takes the operations specified in the SPML requests and executes provisioning actions accordingly. The CMDRA and the SPML Manager are both requesting authorities.

Note: Command line requesting authorities such as CMDRA and SPML Feed can accept input from property files. Requesting authorities' process property files using `java.util.Properties` class. For that reason, certain character such as backslash (\) should be escaped. For example, username parameters in property files should be specified as `Domain\\Username`. For more details on usage requirements, see `java.util.properties` documentation.

Before a Requesting Authority can send requests to the SPML Service, it must be authenticated using HTTP basic client authentication. The Requesting Authority must provide the login credentials of a valid eTrust Admin user.

Important! When the client prompts for the username and password, the Username must include the user's domain name and a backslash in the format `Domain\Username`.

Install SPML

If you intend to use SPML in a FIPS-compliant environment, see the release notes for any installation considerations.

To install the SPML service

1. Locate the Identity Manager installation DVD or other installation media.
2. Start the Product Explorer.
3. Select Install Products, Clients.
4. Select SPML Manager/Server

Answer the questions to provide information about your system.

Uninstall the SPML Service

To un-install SPML, use the Windows Add or Remove Programs option and remove Identity Manager.

Important! This action removes all Identity Manager products.

SPML Service Configuration

During the installation of the Identity Manager SPML Service, you specified a single Provisioning Server. We can now run a Requesting Authority to connect to the SPML Service and start sending provisioning requests targeting that Provisioning Server. But you can also configure the SPML Service to manage multiple Provisioning Servers.

This section explains how to perform functions using the application. It covers topics such as the following:

- Log on to the SPML Configuration Application
- Add a new SPML service
- Modify, rename and delete an existing service

Log On to the SPML Configuration Application

These instructions assume that you have installed the SPML Service on the local computer.

To access and use the SPML Configuration Application, follow these steps:

1. Click Start Menu, Programs, CA, Identity Manager, IM SPML Service Configuration.

The SPML Configuration Login page appears.

2. Log in to SPML Configuration by entering your eTrust Admin server login credentials. A user name with Delegated Administration (DAWI) privileges must be used:

Username

Provisioning Server user with administrator privileges.

Password

Password for this username.

Service

Admin Service to authenticate against.

Domain Name

Enter the domain name to which the username belongs.

Note: The domain name and password are case-sensitive.

3. Click Enter.

You are now logged in to the SPML Service Configuration application.

Add a New SPML Service

To add a new SPML service using the SPML configuration application, follow these steps:

Note: These instructions assume that you have installed the SPML Service on the local computer.

1. Log on to the SPML Configuration application.
2. Enter the following fields in the Admin Service Details form, on the right side of the screen:

Service Name

Specifies a reference name to Provisioning Server service. This name must not appear in the list of available Provisioning Servers on the left side or this will modify the existing service rather than create a new one.

Admin Hostname

Specifies the name of the computer running the Provisioning Server.

Clear Port Number

Specifies the LDAP port number used for communication with the Provisioning Server. By default, this port number is 20389.

SSL/TLS Port Number

Specifies the LDAP TLS port number if you are securing communication with TLS encryption. By default, this port number is 20390.

User SSL/TLS

Select this option. For security reason, the TLS encryption must be used.

3. Click the Add/Modify Service button to save the new service.

Note: If you enter *adminserver* in the Service Name field and *adminserver.yourcompany.com* in the Admin Hostname field then the Requesting Authority client would need to use the URL *http://spmlserver.yourcompany.com:8443/iamspml/spml/adminserver* when connecting to the SPML Server in order to send provisioning requests to this Provisioning Server.

You should replace *spmlserver.yourcompany.com* and *adminserver.yourcompany.com* with the names of the computers on which the SPML Service and the Provisioning Server are running.

Modify an Existing Service

To modify an existing Provisioning Server Service, perform the following steps:

1. Log on to the SPML Service Configuration application.
2. Select the service from the list of Available Admin Services

3. In the Admin Service Details form, make any modifications to the following fields:

Admin Hostname

Clear Port Number

SSL/TLS Port Number

User SSL/TLS

4. Click the Add/Modify Service button.

Note: If you modify the Service Name field you will create a new instance of an SPML service rather than modifying an existing one.

Rename an Existing Service

To rename an existing Provisioning Server Service, perform the following steps:

1. Log on to the SPML Service Configuration application.
2. Select the service from the list of Available Admin Services.
3. Click the Remove the Selected Service button to remove the service from the list.
4. On the form on the right, enter the new name in the Service Name field.
5. Click the Add/Modify Service button.

The new service is added to the list of Available Admin Services with the new name.

Delete an Existing Service

To delete an existing Provisioning Server Service, perform the following steps:

1. Log on to the SPML Service Configuration tool.
2. Select the service from the list of Available Admin Services.
3. Click the Remove the selected Service button.

The service is now deleted.

Configure SSL Support for Tomcat Servers

The Secure Socket Layer (SSL) is a technology that helps ensure the authentication, integrity, and confidentiality of SPML messages. For information on setting up the SSL, see the Configuration HOWTO at <http://jakarta.apache.org/tomcat/>.

Note: The following procedure is provided for reference only. You may want to configure your SSL certificate differently or change your keystore password to one of your own choosing for better security. Also, if you have installed JDK version 1.5, you should refer to <http://jakarta.apache.org/tomcat/> for details.

To install and configure SSL support for Tomcat using a self-signed certificate, perform the following steps:

1. Verify that JDK version 1.4.2_04 is installed by selecting the Add/Remove Programs list in your Control Panel for the program Java 2 SDK, SE v 1.4.2_04.

2. Create a new keystore containing one self-signed certificate by entering the appropriate command from the command prompt.

On Windows systems, you should enter:

```
%JAVA_HOME%\bin\keytool -genkey -alias tomcat -keyalg RSA -keystore  
\path\keystore_filename
```

On UNIX systems, you should enter the following:

```
%JAVA_HOME%/bin/keytool -genkey -alias tomcat -keyalg RSA -keystore  
\path\keystore_filename
```

The keystore creation process begins.

3. Enter the keystore password when prompted.

Note: The default password used by Tomcat is changeit (all lowercase). If preferred, you can specify a custom password, but you must then specify the custom password in the server.xml configuration file also (see Step 8).

The keystore creation process continues.

4. Enter general information for the certificate when prompted. The general information includes company, contact name, and so on. This information displays to users who attempt to access a secure page in your application, so make sure that the information provided here is appropriate.

The keystore creation process continues.

5. Enter the key password when prompted. This password is created specifically for this certificate (not for any other certificates stored in the same keystore file). You must use the same password for this and the keystore password.

A keystore file with a certificate that your server can use is created.

6. Browse to the <Tomcat_installation_directory>\conf\ directory and open the server.xml file in a text editor.
7. Ensure that the SSL Coyote HTTP/1.1 Connector entry is not commented out in the file. The connector information looks similar to the following:

```
<!-- Define an SSL HTTP/1.1 Connector on port 8443 -->
<!--
<Connector className="org.apache.catalina.connector.http.HttpConnector"
    port="8443" minProcessors="5" maxProcessors="75"
    enableLookups="true" acceptCount="10" debug="0" scheme="https"
    secure="true">
    <Factory className="org.apache.catalina.net.SSLServerSocketFactory"
        clientAuth="false" protocol="TLS"/>
</Connector>
-->
```

If the Connector element is commented out, you must remove the comment tags, defined as less than sign, exclamation point, hyphen, hyphen (<!--) and hyphen, hyphen, greater than sign (-->) around it.

8. Configure the SSL Coyote HTTP/1.1 Connector entry to include the keystoreFile and keystorePass attributes for the Factory element.

keystoreFile

Specifies the location where the keystore file is located.

keystorePass

Specifies the keystore (and certificate) password.

The connector information should look similar to the following:

```
<Connector className="org.apache.catalina.connector.http.HttpConnector"
    port="8443" minProcessors="5" maxProcessors="75"
    enableLookups="true" acceptCount="10" debug="0" scheme="https"
    secure="true">
    <Factory className="org.apache.catalina.net.SSLServerSocketFactory"
        keystoreFile="your_keystore_full_path"
        keystorePass="your_keystore_password"
        clientAuth="false" protocol="TLS"/>
</Connector>
```

9. Save the file and close it.

SSL support and self-signed certificates are configured for Tomcat.

10. Restart the Tomcat server.

Configure SPML Client Computer to Support SSL Security

The SPML Web Service requires that the Secure Socket Layer (SSL) be enabled. The SPML clients, the CMDRA, SPML Manager, and SPML Feed must trust the SSL server certificate to communicate with the server.

Note: Third party requesting authorities will need to support SSL to communicate with the SPML Web Service.

To configure the SPML client computer to use SSL security, perform the following steps:

1. Install the SSL certificate to the user's trusted keystore on the computer where the Requesting Authority runs. (By default, the SSL certificate will be added to the .spmlkeystore file in the user's home directory, as determined by the %HOMEPATH% system property.)
 - a. In a web browser, open the following URL:
`https://spmlserver.yourcompany.com:8443`
 - b. Double click on the SSL certificate icon at the bottom right corner of web browser to view the certificate.
 - c. On the Certificate Viewer window, select the Details tab and click Copy to File.
 - d. Save the server certificate.
 - e. Run the following command:

```
<drive>:\<JRE-File-Path>\bin\keytool -import -file <Certificate-File-Path> -  
keystore "%HOMEDRIVE%%HOMEPATH%\spmlkeystore" -storepass changeit -noprompt
```

This command creates a new keystore called .spmlkeystore, located in user's home directory (as determined by "%HOMEDRIVE%%HOMEPATH%"). The batch files that launch the RA clients (SPMLManager, Command Line RA, and SPML Feed) read this file to allow SSL communication.

Note: By default the batch files use the truststore path and password as defined by the keytool command described in step 1e. To use different path and password, variables set in the batch files for each client have to be modified accordingly. For example:

```
set TRUSTSTORE=%HOMEDRIVE%%HOMEPATH%\spmlkeystore  
set TRUSTSTORE_PASSWORD=changeit
```

2. Test the SPML Service with the Command Line RA:
 - a. Open the login.properties file, in the Command Line RA directory, to make sure that HTTPS version of the Server URL is used and user logon details are correct.
 - b. Open the command line prompt.
 - c. From the Command Line RA directory, type:
`RA.batsampleXML\schemaRequest.xml`

CMDRA Commands

CMDRA Command Options

This table shows you the command options that you can use in the CMDRA.

Command	Full Command	Explanation
-c	--check	Check the request is a valid SPML request. This will not send the request to the SPML Server. If you are using the SPML Templating feature, the records will be expanded and the resulting SPML request will be checked.
-e	--explodedOutputFile	Specify a file to contain the exploded request XML output, overwriting any file by that name.

Command	Full Command	Explanation
-f	--propertyFile	Specify a file that contains default command settings. You would create a property file that contains frequently used values to avoid having to specify them manually every time. For example: <pre>mappingFile=C:\SPMLdata\Mapping1.csv templateFile=C:\SPMLdata\ImportUsers.xml dataFile=C:\SPMLdata\Users.csv serverURL=https://spmlserver.yourcompany.com:8443/iamspml/spml/adminserver</pre> You would not typically include the data file in the property file because it is highly variable. The CMDRA looks for the property file <i>login.properties</i>. If your property file is named <i>login.properties</i> you do not need to specify -f. When you specify information in the property file, you should use the full command but remove the two dashes (--).
-h	--help	Display the command line help page that gives you a summary of these commands.
-i	--inputFileNames	Specifies a file to read data/request file names from instead of putting file names on the command line
-m	--mappingFile	Use this to match your Velocity template variables with your data file variables. Typically this is generated using the Save Mapping button in the SPML Manager. You can include multiple mappings that are separated by commas if you include multiple -t options. Note: This must always be used in conjunction with the Data File and the Template File.

Command	Full Command	Explanation
-o	--outputFile	Specify where you want the response from the SPML Service stored. By default, this is written to stdout. This output can be redirected to a file, for example: RA.bat <i>yourparameters</i> > SPMLresponse.text This can contain information other than the raw XML response.
-p	--password	Specify the Provisioning Server password for the user. Therefore, this must be used in conjunction with the user (-u). You do not need to specify the user or password if you are just checking a request.
-q	--quiet	Specify that you want minimum detail in the output.
-R	--explodePerRowOutputFileNames	Specify file that will list the names of files that contain the exploded request XML output, overwriting any file of the same name. One file is created per template per datafile record.
-s	--serverUrl	Specify the SPML server URL that you want to send the request to.
-S	--csvRuntimeStatistics	Name of .CSV file to write time to complete each request
-t	--templateFile	Specify the SPML Template file. Typically you would create the template file using the SPML Manager. If the -R option is included, the -t option can include a comma separated list of filenames. Note: This must always be used in conjunction with the Data File and the Mapping File.

Command	Full Command	Explanation
-u	--user	Specify the Provisioning Server user. This must be used in conjunction with password (-p). You do not need to specify the user or password if you are just checking the request. You must include the domain of the user, for example: YOURDOMAIN\user
-V	--verbose	Include detailed information in the output.
-v	--version	The version and build number of the CMDRA. This must be lowercase.

CMDRA Examples

The first example creates several small requests:

```
Ra.bat -t 01_add_user.xml.vpp,02_modify_user.xml.vpp -R req_file_names.txt -m -,funny_mapping.csv data10.csv
```

- When used with the -R option, the -t template and -m mapping options accept multiple files separated by commas.
- The -R option creates a request file per template per datafile record and the names of resulting files are collected in the named file. So this command creates file names 01_add_user0000000.xml and 02_modify_user0000000.xml through 01_add_user0000009.xml and 02_modify_user0000009.xml.

These files are written to the same directory as the -R file (.) and appended to ./req_file_names.txt.

The second example creates many requests and reports basic performance metrics:

```
Ra.bat -S stats.csv -i req_file_names.txt
```

- The `-i` option takes the name of a file containing names of SPML request files. In this case, the filename is output by the first example.

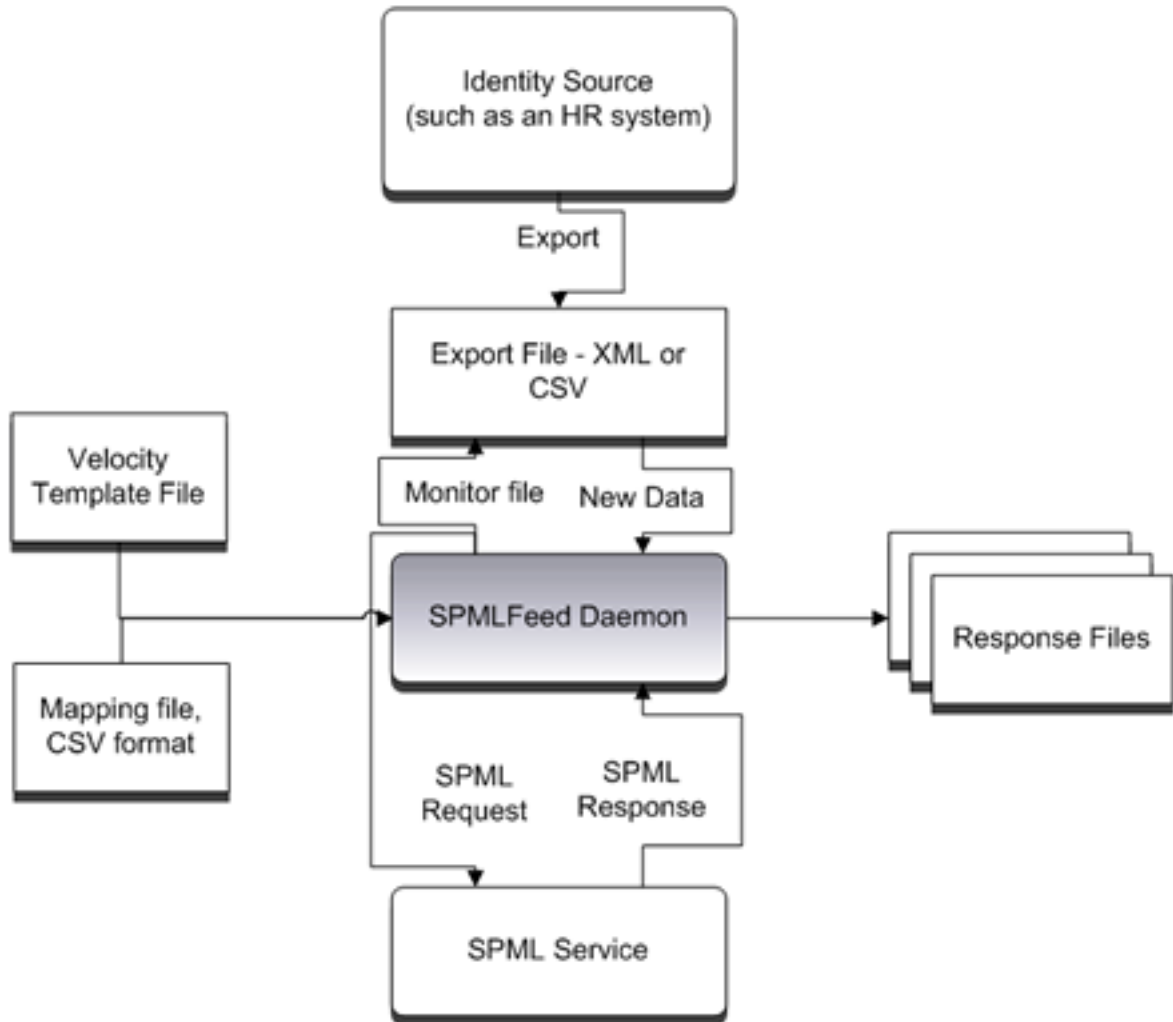
The `-S` option can be included when you submit SPML requests without a template (whether using file names on the command line, or the `-i` option, or both). The option records the execution time in milliseconds for each request and presents summary data after execution.

SPML Feed

SPML Feed is a command line application that you start by using the `feeder.cmd` (on Windows) or `feeder.sh` (on Unix). All command options can be set by using a properties file; some options can be specified on the command line. Options specified on the command line override those in the property file. A property file is used when you include the `-f` or `-propertyFile` command line option.

You can run SPML Feed in standard or daemon mode. In standard mode, the application runs once over the specified input files and then quits. In daemon mode, the application does not terminate. Instead, it periodically wakes up and checks if any input files has been modified. If a file has been modified, it is processed as it would be in standard mode.

The template supplied to the SPML Feed by using the -template option is applied to every record in the input CSV or XML files. The template produces a single SPML add, modify, or delete request.



The template is in the cmdra.zip file. To access it, go to Start, Program CA, Identity Manager, IM SPML Requesting Authority.

SPML Feed Command Options

Options can have three names: the short command line name (-s), the long command line name (--serverURL) and the property file name (serverURL).

-b, --batchSize, batchSize

Integer value that defaults to 1. If this number is greater than 1, requests will be submitted as batch requests, with this value determining how many requests are placed in each batch.

By batching multiple requests together, performance can be improved because less time is spent performing TCP socket and SOAP setup, at the expense of additional memory usage. The memory used depends on the size of your requests, but setting batchSize to several hundred should pose no problem.

Note: batchSize should be always be 1 if your template is for a search or batch request, as these cannot be placed inside a batch request.

-d, --daemon, daemon

Causes the application runs in daemon mode.

-x, --explodeOnly, explodeOnly

The template is exploded using the data files as normal. However, instead of sending the resulting request to the SPML server, it is written to stdout (or a file if output or daemonResponseDir is set).

-h, --help

Display help for the command line switches to stdout.

-i, --inputFileNames, inputFileNames

Specifies a text file. Each line in the text file is used as the name of an input file. This is an alternative method of listing input files on the command line itself. The set of input files is the union of those listed in this file and those listed on the command line.

-l, --logging, logging

Specifies a properties file to configure the Java logging system instead of using the default logging system. See the documentation for the `java.util.logging.LogManager` class for details.

-m, --mappingFile, mappingFile

Specifies a CSV file in a special format that maps parts of the input files to Velocity variable names. The variable names defined in this file can be referenced in the Velocity template file. If there is a mapping to a variable named timestamp, this has a special meaning and is used to determine which records have been changed since the last run.

-o, --outputFile, outputFile

Specifies a file to record the output, overwriting any existing file. If you omit this option, and the daemonResponseDir property is not set, output is written to stdout.

-p, --password, password

Specifies the password to use to authenticate with the SPML server.

-f, --propertyFile

Specifies a property file with the options to use. You can specify any command line option (except -h and -f) in the properties file by setting a property that matches the option's long name without the - prefix. For example, -mapping becomes simply mapping in the properties file. Some options for daemon mode can only be set via a properties file. Any option specified on the command line replaces a setting from the properties.

-q, --quiet, quiet

Causes no output to appear unless a catastrophic failure occurs. In that case, an error message is output to stderr before the program exits.

-s, --serverUrl, serverUrl

The URL of the SPML server to send the request to.

-t, --templateFile, templateFile

Specifies a Velocity template file that can be merged with data from XML or CSV input files to produce an SPML request. The application runs in exploder mode; the input files are either XML or CSV files instead of SPML requests. Each record in the input files is applied to the template (after mapping to variable names via the mapping file) to produce an SPML request that is sent to the server.

Note: the template should not contain `<?xml?>` processing headers.

-u, --user, user

Specifies a user name to authenticate with the SPML Server. For the SPML Server, this option's value should be of the form domain\username.

-v, --verbose, verbose

Outputs additional information to stderr about the application's actions.

-V, --version, version

When the application starts, its name and version number is written to stderr.

Property File Only

timestampFile

Specifies a file used to record when input files have been processed. This is mainly useful in daemon mode to keep track of the latest run times when the daemon is shutdown temporarily or restarted.

daemonResponseDir

Specifies a directory to write SPML responses to when running in daemon mode. The SPML response from each run over an input file by the daemon is written to a new file in this directory. The output files have the same name as the input data file, with a digit appended. For example, if the file test.csv is processed for the first time, the response is written to test.csv.1, the second run to test.csv.2, and so on.

If you omit this option, all responses are written to the file specified with -o, or stdout if neither -o nor -q are present.

daemonSleep

Specifies the length of time in milliseconds to sleep between polling for data file changes. If this parameter is not specified, the length of sleep time is 30 seconds.

Flow of the SPML Feed Command

The flow of the SPML Feed command is:

1. This command is invoked with these parameters:
 - Mapping file
 - Velocity template that produces a valid SPML request.
 - Data files to watch
 - Output file or output directory
 - Length of time to sleep between polling for file data file changes.
2. When a data file changes, the running daemon begins processing the file on the next poll.
3. The data file is locked to prevent writes.

4. The daemon reads the data file one record at a time.
 - If the mapping file has an entry mapping to the special value timestamp, that field will be retrieved from the record. If the record's timestamp is earlier than the last time the file was processed, the record has not changed and is skipped.

Note: The timestamp should be specified in GMT time zone and YYYY-MM-DD HH:MM:SS format, such as 2006-02-14 21:02:03. If the timestamp is not in this format, or is absent, the request is skipped.
 - If the record is not skipped, it is loaded into a Velocity context and merged with the template to produce an SPML request.
 - If the mapping file does not have an entry mapping to the special value timestamp, the daemon submits all requests from the data file without exception.
5. The SPML requests are submitted to the SPML Service.
6. The response from the SPML Service is classified and appended to the appropriate output files.

The SPML Feed can use the Velocity templates created for the SPML Manager and CMDRA applications if the batchSize argument is set to 1. Ideally, you should modify templates to match SPML Feed requirements, which are less strict.

Using the SPML Manager's Templating Functionality

The SPML Manager is a graphical user interface that lets administrators create and execute SPML provisioning requests. You can design provisioning requests using the SPML Manager then view the SPML Request in XML format.

Note: The SPML Manager is an unsupported technology preview.

Download the SPML Manager

1. Download the SPML Manager from the following location:
`https://spmserver.yourcompany.com:8443/iamspml/download/techpreview/SPMLManager.zip`
2. Unzip SPMLManager.zip to your hard disk.
3. To launch the SPML Manager, navigate to the SPMLManager folder and double-click SPMLManager.bat.

Create an SPML Template Request

To create an SPML Template request, perform the following steps:

1. Create an XML file containing some sample data.
You can find a sample in the SPMLManager\sample Templates directory.
2. Using the SPML Manager, connect to an SPML service.

3. Use one of the tabs (Add, Modify, Delete, or Extended) to construct an SPML request that conforms to the schema of the SPML service.

The name given to the Exploder Velocity context is REC_, so whenever you want to refer to a variable in the data file, use the syntax `${REC_.variable}`.

To hard-code a constant in the generated requests, type the data you want into the attribute fields.

4. Click the Raw XML tab to see the SPML Request in XML format.
5. Click the Add to Batch Request button when you are happy with your simple variable replacement.

Note: The Add to Batch Request button is only available when viewing the tab in Request mode and is therefore not available in Raw <XML> mode.

6. Use the Batch tab to see the addition to batch.

A batch request can contain as many individual requests as you like.

7. Save the SPML Batch Request Template to a file.
8. Click the Template tab and load your simple SPML Template from your saved Batch Request File.
9. Click the Raw <XML> tab to edit your SPML Template:
10. In the XML code, insert a velocity directive for each loop at the point in your batch request where you want the request to cycle through loading each row in the CSV into the context and put a #end statement where you want the cycling to end. The format of the syntax for each loop is `#foreach ( $REC_ in $RECS_`.

11. Click Save Template to save your modified template file.
12. Load the XML data file that will fill in the data in your request.
13. Click the Save button if you want to Save the resulting batch request to a file for inspection or click Submit if you want to submit the resulting Batch Request to the SPML Service.

For each XML record in the data file, a corresponding SPML Request will be generated to initiate a provisioning operation inside the eTrust Admin server.

The example files used in this tutorial are in the SPMLManager/sampleTemplates/simple directory.

Using Velocity Templates

The SPML Manager, SPML Feed, and CMDRA use the same templating system, which parses references to variables and performs data transformations.

List Templating Variables

The templating system deduces which variables are ArrayLists by parsing references to them in the Velocity template you provide. The templating system looks for an attribute references or method invocations against the variable which show that it should be bound to a java.util.ArrayList (which can have 0, 1 or more values) rather a single value. The SPML Manager and CMDRA can use this capability.

All the read-only methods on the class java.util.ArrayList are looked for:

- isEmpty
- get
- size
- contains
- indexOf
- lastIndexOf
- subList
- iterator
- listIterator

In mapping file entries, such list variables have the suffix [] after their names. For instance the variable `comments` is a single valued variable, but `comments[]` would be a list.

For example references, see the `sampleTemplates\simple\template.xml.vpp` template, including the `$comments` variable. Also, included are example mapping files: `map_csv_datafile.csv` against the CSV datafile and `map_xml_datafile.csv` against the XML datafile.

Data Transformations

SPML Templating offers several routines for manipulating data from a unique or proprietary format into the format required by the SPML Service which is generally the same as the standard XML Schema Data Types (see <http://www.w3.org/tr/xmlschema-2> for more information). These tools are provided by the Velocity Tools project which is in the SPML Requesting Authority classpath and therefore available for reference in your SPML Template.

SPML Manager, SPML Feed, and Requesting Authority.

Useful tools for manipulating and transforming data inside of your SPML Template include the following:

- **Date Tool:** A tool for manipulating and formatting dates.
- **Math Tool:** A tool for performing floating point math.
- **Number Tool:** A tool for formatting numbers.
- **Iterator Tool:** A tool to use with `#foreach` loops. It wraps a list to let the designer specify a condition to terminate the loop, and reuse the same list in different loops.
- **Render Tool:** A tool to evaluate and render arbitrary strings of VTL (Velocity Template Language).

The templating system loads data from XML and CSV files in string format. You can use these tools to convert data in your template to the type required for the operation you want to perform with the data.

Example Data Transformation

This example takes a numeric value which has been converted to a string inside the templating system and then converts it into an integer value to perform preprocessing on the data before sending it to the SPML Service.

In this example, the imported CSV file contains only three pieces of data.

- username,expirydate,priority
- user3,20101001,999
- user4,20101005,333

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<batchRequest onError="urn:oasis:names:tc:SPML:1:0#resume"
processing="urn:oasis:names:tc:SPML:1:0#sequential"
execution="urn:oasis:names:tc:SPML:1:0#synchronous"
xmlns="urn:oasis:names:tc:SPML:1:0">
#foreach ( $REC_ in $RECS_ )
    #set ( $userhandle =
"User=${REC_.username},Domain=YOUR_USER_DOMAIN,Server=Server")
    ''
    #set ( $datetimeobject =
$date.toDate('yyyyMMdd',$ {REC_.expirydate}))
    #set ( $formatdate = $date.format('yyyy-MM-dd', $datetimeobject))
    #set ( $formattime = $date.format('H:m:s', $datetimeobject)) ''
    <addRequest requestID="batchAdd${REC_.INDEX}">
        <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
            <id>$userhandle</id>
        </identifier>
        <attributes>
            <attr name="accountId">
                <ns1:value
xmlns:ns1="urn:oasis:names:tc:DSML:2:0:core">${REC_.username}</ns1:value>
            </attr>
            <attr name="roleHandles">
                <ns2:value
xmlns:ns2="urn:oasis:names:tc:DSML:2:0:core">Role=ntRole,Domain=YOUR_USER_DOM
AIN,Server=Server</ns2:value>
            </attr>
            <attr name="enableDate">
                <ns3:value
xmlns:ns3="urn:oasis:names:tc:DSML:2:0:core">'${formatdate}T${formattime}'
            </ns3:value>
            </attr>
        </attributes>
    </addRequest>
    ''#if($math.toInteger(${REC_.priority}) > 500)''
```

```
<extendedRequest xmlns="urn:oasis:names:tc:SPML:1:0">
  <operationalAttributes/>
  <providerIdentifier providerIDType="urn:oasis:names:tc:SPML:1:0#URN">
    <providerID/>
  </providerIdentifier>
  <operationIdentifier
operationIDType="urn:oasis:names:tc:SPML:1:0#GenericString">
    <operationID>User-SyncWithRolesAddAccounts</operationID>
  </operationIdentifier>
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>$userhandle</id>
  </identifier>
  <attributes/>
</extendedRequest>
#end
#end
</batchRequest>
```

This request formats a proprietary data format into a format that eTrust Admin can understand. The results of this request are:

- user3 is created with an XSD Enable Date of 2010-10-01T0:0:0 that has been converted from the yyymmdd format of 20101001
- user3 has been considered a high priority case and has synced with the NT role immediately upon creation to create accounts. This is because user3 had a high priority of 999 which has been evaluated to see if it was greater than 500.
- user4 is created with an XSD Enable Date of 2010-10-05T0:0:0 that has been converted from the yyymmdd format of 20101005
- user4 has not been considered a high priority case for account creation because their priority code was 333 which is less than 500. Their account will not be created until a later stage, perhaps when a nightly sync operation occurs.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<batchRequest onError="urn:oasis:names:tc:SPML:1:0#resume"
processing="urn:oasis:names:tc:SPML:1:0#sequential"
execution="urn:oasis:names:tc:SPML:1:0#synchronous"
xmlns="urn:oasis:names:tc:SPML:1:0">
  <addRequest requestID="batchAdd0">
    <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
      <id>User=user3,Domain=YOUR_USER_DOMAIN,Server=Server</id>
    </identifier>
    <attributes>
      <attr name="accountId">
        <ns1:value
xmlns:ns1="urn:oasis:names:tc:DSML:2:0:core">user3</ns1:value>
        </attr>
        <attr name="roleHandles">
          <ns2:value
xmlns:ns2="urn:oasis:names:tc:DSML:2:0:core">Role=ntRole,Domain=YOUR_USER_DOM
AIN,Server=Server</ns2:value>
          </attr>
          <attr name="enableDate">
            <ns3:value
xmlns:ns3="urn:oasis:names:tc:DSML:2:0:core">' '2010-10-
01T0:0:0' ' </ns3:value>
            </attr>
          </attributes>
        </addRequest>
        <extendedRequest>
          <operationalAttributes/>
          <providerIdentifier providerIDType="urn:oasis:names:tc:SPML:1:0#URN">
            <providerID/>
          </providerIdentifier>
          <operationIdentifier
operationIDType="urn:oasis:names:tc:SPML:1:0#GenericString">
            <operationID>' 'User-SyncWithRolesAddAccounts' ' ' </operationID>
          </operationIdentifier>
          <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
            <id>' 'User=user3,Domain=YOUR_USER_DOMAIN,Server=Server' ' </id>
          </identifier>
          <attributes/>
        </extendedRequest>
        <addRequest requestID="batchAdd1">
          <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
            <id>User=user4,Domain=YOUR_USER_DOMAIN,Server=Server</id>
          </identifier>
```

```

    <attributes>
      <attr name="accountId">
        <ns4:value
xmlns:ns4="urn:oasis:names:tc:DSML:2:0:core">user4</ns4:value>
      </attr>
      <attr name="roleHandles">
        <ns5:value
xmlns:ns5="urn:oasis:names:tc:DSML:2:0:core">Role=ntRole,Domain=YOUR_USER_DOM
AIN,Server=Server</ns5:value>
      </attr>
      <attr name="enableDate">
        <ns6:value
xmlns:ns6="urn:oasis:names:tc:DSML:2:0:core">' '2010-10-
05T0:0:0' ' ' </ns6:value>
      </attr>
    </attributes>
  </addRequest>
</batchRequest>

```

Retrying SPML Requests

You can configure certain requests to be retried on failure. Request retrying is attempted if all of the following are true:

- The request is flagged for asynchronous execution.
- The object on which the request is acting at the time of failure resides on a remote endpoint system, such as an account/container/native group.
- The request is causing a change and not a query.
- The failure occurs after a request has reached the Provisioning Server. It is the client's responsibility to retry if either:
 - The communication channels between the client and the web server on which the SPML server is executing is broken.
 - The communication channel between the SPML server and Provisioning Server is broken.
- The failure is a soft failure between the Provisioning Server and the targeted endpoint system.

Note: Batch requests will not support retrying on their constituent sub-requests.

On successful completion or hard failure, the standard SPML success/failure conditions are returned in the status response's result.

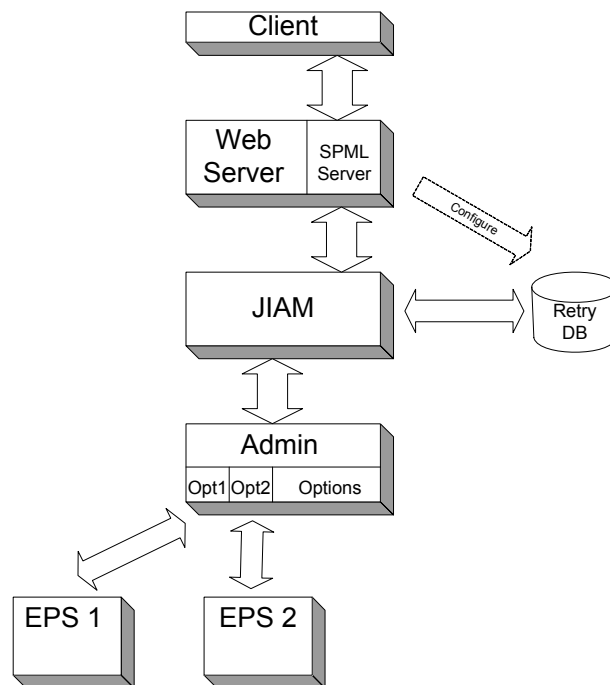
The SPML server is involved in the configuration of the retry persistence mechanism in JIAM but only JIAM actually adds and deletes records from it.

Retry Architecture

You can configure the SPML service to retry asynchronous requests that act on objects residing on Endpoint Systems which cannot be contacted due to a transient failure. The flow of retries follows the standard asynchronous request processing in SPML.

- The client submits an asynchronous request to the SPML server with an additional operational attribute indicating that retries should be attempted on failure.
- If the request is processed without error, the request is marked as successfully processed.
- If the request's processing terminates with a hard failure, or its target object is not considered retrievable by JIAM, an immediate failure results.
- If the request's processing terminates with a soft failure and the client has flagged the request as retrievable, JIAM stores the request in the retry database and attempts to process it again after a configured interval. Prior to storage the current operation is simplified by removing any sub-operations which succeeded, so that only the sub-operations which failed due to soft failures will be retried.
- If a request has been retried a configured maximum number of times, then it is considered to have suffered a hard failure and JIAM discontinues the retries.

The following diagram gives an overview of the components of the retries:



Note that the SPML server is involved in the configuration of the retry persistence mechanism in JIAM but only JIAM actually adds and deletes records from it.

Retry Configuration Files

Interface

Two operational attributes control the SPML retry capability:

- `calamRetry` can be passed in with an asynchronous request, requesting that retry functionality is activated where supported by JIAM. `calamRetry` is operational attribute and its value can be true or false.
- `calamRetryDetails` is returned when an SPML status request targeting a retried request is received. It provides status information to the user.

For an example of a request flagged for retry, see the appendix “Sample SPML Requests.”

Hivemodule-plugin-deploy.xml

The file `hivemodule-plugin-deploy.xml` in the SPML deployment directory includes configuration items:

- A new database called `db_spml_retry` configured by the `retryBasicDataSource` service-point, and advertised to JIAM via the `retryPersister` service-point. This database (as well as the pre-existing `db_qrtz` and `db_delegate`) should not grow and shrink as requests are processed and time out.

If a problem occurs in this area, the databases can be truncated by shutting down the SPML server and simply deleting their directories (the SPML server will automatically create new blank directories on start-up).

- The `retryMaximumCount` and `retryDelayMinutes` settings which control the JIAM retry behavior using the `JIAMService` class, which is also provided with the `retryPersister`.

`retryMaximumCount` tells the SPML server the maximum number of times it should retry an operation before giving up and returning an error. When a retry is needed, the SPML server waits before doing the retry, to give the network or server failure a chance to correct itself. The time it waits before retrying is controlled by the `retryDelayMinutes` setting.

- The `scheduledHoldingIntervalMinutes` setting dictates the time that status requests targeting an asynchronous request can be submitted after its processing is completed, and consequently applies to retried requests too.

Any manual changes made to a `hivemodule-plugin-deploy.xml` file from a previous release will need to be reapplied to the file for this release.

To cancel a retried request, submit a SPML cancel request referring to its identifier. If a retry attempt has begun, it will continue, but no further retry attempts will be made.

Note: Retry functionality is not supported on SPML modify requests that change an object's Distinguished Name (DN).

Access Credentials

Access credentials for the databases used by the SPML server are stored in the `spml_quartz.properties` and `hivemodule-plugin-deploy.xml` configuration files. Each database supports two different styles of URLs in their datasources:

- `jdbc:hsqldb:file:<dir>\<db>\<db>` : databases can only be accessed by the SPML server process itself (the default).
- `jdbc:hsqldb:hsqldb://localhost/<db>` : allows SQL access to the database by other processes.

Configure Retry for a Request

The process of configuring an SPML request to be retried is the following:

1. In the SPML Manager or Command-line RA, you submit a request for asynchronous execution including the `calamRetry` operational attribute. This attribute can be referenced in Java as `com.ca.commons.spml.IAMSpmlUtil.CA_IAM_ATTR_RETRY`.
2. The SPML server schedules the request for immediate execution. If the processing is completely successful or any sub-operation fails with a hard failure, a success or failure SPML response is stored.

Otherwise if one or more sub-operations fail with a soft failure, the SPML server will return `true` from the `actionFailed()` method of the `com.ca.iam.spml.ProcessingDetails` instance assigned as the `IAMCommitObserver` for the current session, which informs JIAM that there is an interest in retrying.

3. You can track processing of an asynchronous request by submitting an SPML status request quoting the request ID. The status eventually changes to `complete`, signifying either complete success or a non-retriable failure. The non-retriable state could be due to the retry limit being exhausted as set by `retryMaximumCount` in `hivemodule-plugin-deploy.xml`.
 - a. Results for asynchronous requests are cached for a configurable period after their processing completes (refer `scheduledHoldingIntervalMinutes` in `hivemodule-plugin-deploy.xml`)
 - b. When the status of a request which is being, or has been, retried is queried, the operational attribute `calamRetryDetails` appears in its operational attributes and provides a rough summary of progress suitable for a human reader.
4. JIAM then analyses the operation that failed and determines if it does indeed support retrying for it. If retrying is supported, JIAM will remove any successful sub-operations and use the `HSQLDBPersister` configured by the SPML server to save the retry operation to the database named `db_spml_retry`. The `IAMSession.commit()` call then completes and a pending SPML response is returned to the RA that submitted the request.

5. The JIAM retry subsystem periodically retries the request based on the `retryDelayMinutes` setting in `hivemodule-plugin-deploy.xml`. assuming `retryMaximumCount` limit is not exhausted.
 - a. If a soft failure is encountered, the retry process repeats at step 4.
 - b. If a hard failure is encountered or the limit is exhausted, the retry sequence terminates and the SPML server is informed via the `failed()` method of the registered `QueueObserver`. Future status requests targeting the retried request will report failure.
 - c. If no failures are encountered, JIAM informs the SPML server that the request was processed successfully via the `completed()` method of the registered `QueueObserver`. Future status requests targeting the retried request will report success.

Chapter 11: Sample SPML Requests

This appendix describes the sample SPML requests that a Requesting Authority can use to send provisioning requests to the SPML Service.

For a detailed description of the format of these requests, see the SPML v1.0 specification at:

<http://www.oasis-open.org/committees/download.php/3032/cs-pstc-spml-core-1.0.pdf> (<http://www.oasis-open.org/committees/download.php/3032/cs-pstc-spml-core-1.0.pdf>)

This section contains the following topics:

[Request Execution Types](#) (see page 161)

[Request Types](#) (see page 162)

[Global Settings](#) (see page 174)

[Account Containers](#) (see page 176)

[Complex Attributes](#) (see page 178)

[Request Retries](#) (see page 180)

[Propagate Global User Changes](#) (see page 180)

[Escaping Special Characters in Object Identifiers](#) (see page 182)

[Escaping Special Characters in Search Filters](#) (see page 182)

Request Execution Types

The Identity Manager SPML Service supports both synchronous and asynchronous request models.

By default if no execution attribute is set with a request to the SPML Service the request will be treated as having a synchronous execution mode.

If *synchronous* is specified, any request that you send must be completed and a response sent back before the next request can be sent. This can sometimes cause delays.

If *asynchronous* is specified as the execution mode, the SPML Service will schedule the request to be executed asynchronously and return immediately with a unique request ID. The Requesting Authority can later look up the corresponding result of the request by specifying the request ID in a Status Request. Any pending asynchronous request can be canceled by specifying the request ID from the asynchronous request in a Cancel Request.

Request Types

Add Request

The add request is used by a Requesting Authority to create new entity instances such as User, Role, Group, Profile, Policy, EndPoint, or Account objects.

Fields in an Add Request

Objects may contain mandatory fields that must be populated in order for the object to be created.

An add request contains the following fields:

identifier

Specifies the ID of the new object to be created

attributes

(Optional) Specifies initial values for some of the attributes as appropriate

Example of an Add Request

The following request creates a new User object with the unique identifier User=_spml_user,Domain=EXAMPLE_DOMAIN,Server=Server:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<addRequest xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsm1="urn:oasis:names:tc:DSML:2:0:core">
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>User=_spml_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
  </identifier>
  <attributes>
    <attr name="accountId">
      <dsm1:value>_spml_user</dsm1:value>
    </attr>
    <attr name="suspended">
      <dsm1:value>true</dsm1:value>
    </attr>
  </attributes>
</addRequest>
```

Batch Request

The batch request collates multiple SPML operations into a single request.

Example of a Batch Request

The following batch request executes these SPML operations in this order:

- Adds the user *_spml_user*
- Modifies *_spml_user* to update the comments attribute
- Deletes the *_spml_user* object
- Performs a CheckSync operation on the user *administrator*

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<batchRequest execution="urn:oasis:names:tc:SPML:1:0#synchronous"
onError="urn:oasis:names:tc:SPML:1:0#resume"
processing="urn:oasis:names:tc:SPML:1:0#sequential"
xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsm1="urn:oasis:names:tc:DSML:2:0:core">
  <addRequest requestID="batchAdd">
    <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
      <id>User=_spml_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
    </identifier>
    <attributes>
      <attr name="accountId">
        <dsml:value>_spml_user</dsml:value>
      </attr>
    </attributes>
  </addRequest>

  <modifyRequest requestID="batchModify">
    <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
      <id>User=_spml_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
    </identifier>
    <modifications>
      <modification name="comments" operation="replace">
        <dsml:value>new comments</dsml:value>
      </modification>
    </modifications>
  </modifyRequest>

  <deleteRequest requestID="batchDelete">
    <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
      <id>User=_spml_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
    </identifier>
  </deleteRequest>
```

```
<extendedRequest requestID="batchExtended">
  <providerIdentifier providerIDType="urn:oasis:names:tc:SPML:1:0#URN">
    <providerID>urn:ca.com:etrust:iam</providerID>
  </providerIdentifier>
  <operationIdentifier
operationIDType="urn:oasis:names:tc:SPML:1:0#GenericString">
    <operationID>User-CheckSync</operationID>
  </operationIdentifier>
  <attributes>
    <attr name="IAMUser">
<dsml:value>User=administrator,Domain=EXAMPLE_DOMAIN,Server=Server</dsml:value>
    </attr>
  </attributes>
</extendedRequest>
</batchRequest>
```

Cancel Request

The cancel request allows a client to request the cancellation of an asynchronous request from the SPML Service.

Example of a Cancel Request

For example, a previously-sent asynchronous with the request ID A4DF567HGD can be cancelled with the following request:

```
<cancelRequest requestID="A4DF567HGD" xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core"/>
```

Delete Request

Use the delete request to delete any object that is available through the Provisioning Server, except a domain or namespace object. Domain and namespace objects cannot be created or deleted through SPML requests.

Fields in a Delete Request

A delete request contains only one field:

identifier

Specifies the ID of the object to be deleted

Example of a Delete Request

The following request deletes the *_spml_user* object from the Provisioning Server:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<deleteRequest xmlns="urn:oasis:names:tc:SPML:1:0">
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>User=_spml_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
  </identifier>
</deleteRequest>
```

Extended Request

Use the extended request to perform actions that are unique to the Provisioning Server, such as explore, correlate, sync, and checkSync.

Fields in an Extended Request

An extended request contains the following fields:

operationIdentifier

Specifies the type of the extended operation performed.

identifier

Identifies the object that the extended operation is to be applied to.

attributes

Passes parameters specific to the extended operation. The parameters required by an extended operation are in the core Schema Response.

Example of an Extended Request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<extendedRequest xmlns="urn:oasis:names:tc:SPML:1:0"
  xmlns:dsm1="urn:oasis:names:tc:DSML:2:0:core">
  <providerIdentifier providerIDType="urn:oasis:names:tc:SPML:1:0#URN">
    <providerID></providerID>
  </providerIdentifier>
  <operationIdentifier
    operationIDType="urn:oasis:names:tc:SPML:1:0#GenericString">
    <operationID>User-CheckSync</operationID>
  </operationIdentifier>
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>User=Administrator,Domain=EXAMPLE_DOMAIN,Server=Server</id>
  </identifier>
  <attributes></attributes>
</extendedRequest>
```

Extended Request Types

Request Type	Function
Account-CheckSync	Check an account against its assigned policies for out-of-sync attributes.
Account-ForcedDelete	Delete this account and clear all references to it.
Account-Relocate	Move the selected account to the correct container as specified by a given policy.
Account-SyncWithPolicies	Synchronize an account with its assigned policies.
Container-CheckAccountSync	Check whether the accounts in a container need to be synchronized with associated policies.
Container-Correlate	Perform a correlation on an endpoint or on a container of a hierarchical endpoint. Scope The search scope. ONELEVEL_SCOPE is represented by the integer 1 and SUBTREE_SCOPE is represented by the integer 2. ONELEVEL_SCOPE Correlates one level the managed accounts on an end point system with the Global Users. SUBTREE_SCOPE Correlates all managed accounts on an end point system with Global Users. CreateUserAsNeeded Use "true" to create Global Users as needed.
Container-Explore	Perform an explore operation on an endpoint or a container of a hierarchical endpoint. Scope The search scope. ONELEVEL_SCOPE is represented by the integer 1 and SUBTREE_SCOPE is represented by the integer 2. ONELEVEL_SCOPE Searches one level for managed objects on the given Container. SUBTREE_SCOPE Searches for all managed objects on the given Container.

Request Type	Function
Container-SyncAccountsWithPolicies	Synchronize the accounts in a container with their assigned policies.
Container-UpdateUserFields	Update the global users' attributes with their correlated accounts' attributes, according to the attribute mappings defined in the defaultUserUpdateMap field of the container object. Scope The search scope. ONELEVEL_SCOPE is represented by the integer 1 and SUBTREE_SCOPE is represented by the integer 2. ONELEVEL_SCOPE Performs the updateUserFields operation for the accounts directly below the given container. SUBTREE_SCOPE Performs the updateUserFields operation for all accounts of the given container.
EndPoint-IncludeContainer	Bring a top-level container into the database (but not its contents). This is required for the exploration operation to work on some hierarchical endpoints such as ADS endpoints where the normal ONELEVEL exploration does not add the top-level container to the eTrust Admin database. This is also useful to manage only a portion of the hierarchical endpoint with the Provisioning Server while the remaining portion is completely hidden to Provisioning Server users. ContainerName The name of the container to include. ContainerType Option-specific types of containers. See the JIAM OptionDescriptor Javadoc for available container types for each option

Request Type	Function
Group-ListMembers	Search for Global Users that are members of this Group. Scope The search scope. ONELEVEL_SCOPE is represented by the integer 1 and SUBTREE_SCOPE is represented by the integer 2. ONELEVEL_SCOPE Searches one level for group members. SUBTREE_SCOPE Searches recursively through all nested child groups for members. UserNameMatchString The string to match the user name against. Use null to return all the users of this group. CountLim The maximum number of users to return. If 0, return all entries that satisfy the above matching expression.
Policy-CheckAccountSync	Check the accounts against a given policy.
Policy-ForcedDelete	Delete this policy and clear all references to it.
Policy-RelocateAccounts	Move the accounts associated with a policy to the correct containers.
Policy-SyncAccounts	Synchronize the accounts associated with a policy.
Role-CheckAccountSync	Check whether the accounts of multiple users need to be synchronized against the policies assigned to this role.
Role-CheckUserSync	Check whether the users of a role need to be synchronized with their associated policies.
Role-DeleteWithPolicies	Delete the role and all associated policies. ForcedDelete Set to <i>true</i> to delete the role and clear all references to it as well as associated Policies.
Role-ForcedDelete	Delete this role and clear all references to it.
Role-SyncAccountsWithPolicies	Synchronize the accounts of multiple users against the policies assigned to a role.

Request Type	Function
Role-SyncUsers	Synchronize the users of a role with all assigned policies.
User-CheckAccountSync	Check whether the accounts of a user need to be synchronized with associated policies.
User-CheckSync	Check whether the user requires synchronization with associated Roles.
User-DeleteWithAccounts	Delete the user and all associated accounts. ForcedDelete Set to <i>true</i> to delete the user and clear all references to it as well as associated accounts.
User-ForcedDelete	Delete this user and clear all references to it.
User-GeneratePassword	Generate a random password that conforms to the password quality rules for a global user.
User-RequestPasswordReset	Register a user's password reset request.
User-SyncAccountsWithPolicies	Synchronize the accounts of a user with their assigned policies.
User-SyncWithRolesAddAccounts	Synchronize a user with roles and create accounts.
User-SyncWithRolesDeleteAccounts	Synchronize a user with roles and delete accounts.

Modify Request

Use the modify request to update all objects provisioned by the SPML server including the namespace and domain objects.

Fields in a Modify Request

A modify request contains the following fields:

identifier

Identifies the object to be modified

modifications

Lists the attributes to be modified. Attribute values can be added, deleted, or replaced as specified in the *operation* flag.

Example of a Modify Request

The following request sets the *comments* attribute of User *_spml_user* to the string *new comment*.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<modifyRequest xmlns="urn:oasis:names:tc:SPML:1:0"
  xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>User=_spml_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
  </identifier>
  <modifications>
    <modification name="comments" operation="replace">
      <dsml:value>new comment</dsml:value>
    </modification>
  </modifications>
</modifyRequest>
```

Propagate Global User Changes

Modifications made to global user attributes can be propagated to the global user's accounts, by adding *syncAccounts* attribute to the modify request and setting it to true. The same rule applies if you modify global user complex attributes such as address.

Note: By default the SPML manager does not display the *syncAccounts* attribute in the Modify tab for any object except from the global user. To propagate changes made to global users complex attributes, add a field by clicking New Modification. Then, specify the name of the field to be *syncAccounts* and set it to true.

Example of a Modify/Propagate Request

```
<?xml version="1.0" encoding="UTF-8"?>
<modifyRequest xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
<operationalAttributes></operationalAttributes>
<identifier type="urn:oasis:names:tc:SPML:1:0#DN">
<id>User=_spml_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
</identifier>
<modifications>
<modification name="comments" operation="replace">
<dsml:value>new comment</dsml:value>
</modification>
<modification name="syncAccounts" operation="replace">
<dsml:value>true</dsml:value>
</modification>
</modifications>
</modifyRequest>
```

Example of a Modify/Propagate Complex Attribute Request

```
<?xml version="1.0" encoding="UTF-8"?>
<modifyRequest xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
<operationalAttributes></operationalAttributes>
<identifier type="urn:oasis:names:tc:SPML:1:0#DN">
<id>address@User=_spml_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
</identifier>
<modifications>
<modification name="city" operation="replace">
<dsml:value>new city</dsml:value>
</modification>
<modification name="syncAccounts" operation="replace">
<dsml:value>true</dsml:value>
</modification>
</modifications>
</modifyRequest>
```

Schema Request

Use the schema request to exchange provisioning schema between the Requesting Authority and SPML Service.

The schema request is used by the Requesting Authority to determine the specific data structures and extended operations that the SPML Service provides access to.

The core eTrust SPML Service schema is identified by the provider identifier *urn:ca.com:etrust:iam* and schema identifier *core*.

Example of a Schema Request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<schemaRequest xmlns="urn:oasis:names:tc:SPML:1:0">
  <providerIdentifier providerIDType="urn:oasis:names:tc:SPML:1:0#OID">
    <providerID>urn:ca.com:etrust:iam</providerID>
  </providerIdentifier>
  <schemaIdentifier schemaIDType="urn:oasis:names:tc:SPML:1:0#GenericString">
    <schemaID>core</schemaID>
  </schemaIdentifier>
</schemaRequest>
```

Search Request

Use the search request to read the attributes of objects provisioned by the Provisioning Server.

Each object is uniquely identified by an ID, which is similar to an LDAP distinguished name. For example, the following identifier is the unique identifier representing the EXAMPLE_DOMAIN domain:

```
"Domain=EXAMPLE_DOMAIN,Server=Server"
```

The following identifies the user *Administrator* in this domain:

```
"User=Administrator,Domain=EXAMPLE_DOMAIN,Server=Server" .
```

A search request allows you to look up objects in a container. In particular, a domain can contain the following objects:

- User
- Group
- Dynamic Group
- Profile
- Password Profile
- Role
- Child Domain
- Namespace

A namespace contains policies and end points, and an end point contains accounts and account containers.

Search Filters

Use an LDAP filter in the search request to identify the objects that you wish to return.

For example, "(name=*)" would list all the objects in this domain.

You can combine several expressions to form a sophisticated filter such as "(&(name=admin*)(role=*newrole*))" in accordance with the LDAP filter search described in RFC 2254.

Fields in a Search Request

A search request contains the following fields:

searchBase

Specifies the starting point for the search operation using the ID string of the container object

filter

Specifies the search criteria

attributes

Lists the attributes to be returned in the search response

Example of a Search Request

The following search request queries the EXAMPLE_DOMAIN to list all objects and return the name and description attributes for all of these objects:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<searchRequest xmlns="urn:oasis:names:tc:SPML:1:0"
  xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
  <searchBase type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>Domain=EXAMPLE_DOMAIN,Server=Server</id>
  </searchBase>
  <filter>
    <dsml:equalityMatch name="name">
      <dsml:value>*</dsml:value>
    </dsml:equalityMatch>
  </filter>
  <attributes>
    <dsml:attribute name="name"></dsml:attribute>
    <dsml:attribute name="description"></dsml:attribute>
  </attributes>
</searchRequest>
```

Status Request

The Requesting Authority uses a Status Request to query the processing status of an asynchronous operation.

Example of a Status Request

The example below is for requesting the status of a previously-sent asynchronous with the request ID "A4DF567HGD".

```
<statusRequest requestID="A4DF567HGD"
xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core"/>
```

Global Settings

Global settings are settings that affect the Provisioning Server Domain. Global settings are changed at the corporate level for entire company. Some of the properties that can be defined in global settings include

- Enabling and disabling self-authentication preferences
- Setting the number of questions
- Setting the number of optional fields

In Provisioning Manager, these settings can be seen from the System (task frame), Global Properties.

Example: Search for Attributes Defined in Global Settings

Here is an example of how to search for attributes defined in global settings.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<searchRequest xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
  <searchBase type="urn:oasis:names:tc:SPML:1:0#DN">
    <id></id>
  </searchBase>
  <filter>
    <dsml:equalityMatch name="name">
      <dsml:value>*</dsml:value>
    </dsml:equalityMatch>
  </filter>
  <attributes>
    <dsml:attribute name="selfAuthEnabled"></dsml:attribute>
    <dsml:attribute name="numberSelfAuthQuestions"></dsml:attribute>
    <dsml:attribute name="numberOptionalSelfAuthProperties"></dsml:attribute>
  </attributes>
</searchRequest>
```

Note: You should not specify a value for the searchBase field. If you are doing this search from the SPML Manager you will need to leave the searchBase field empty.

Example: Modify Attributes in Global Settings

Here is an example of how to modify attributes defined in global settings. Note that the ID string "Server=Server" identifies the global settings object.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<modifyRequest xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsm1="urn:oasis:names:tc:DSML:2:0:core">
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>Server=Server</id>
  </identifier>
  <modifications>
    <modification name="autoGenerateUIDs" operation="replace">
      <dsm1:value>true</dsm1:value>
    </modification>
    <modification name="autoGenerateUIDsForNewUsers" operation="replace">
      <dsm1:value>true</dsm1:value>
    </modification>
    <modification name="numberOptionalSelfAuthProperties"
operation="replace">
      <dsm1:value>5</dsm1:value>
    </modification>
    <modification name="numberSelfAuthQuestions" operation="replace">
      <dsm1:value>4</dsm1:value>
    </modification>
    <modification name="selfAuthEnabled" operation="replace">
      <dsm1:value>1</dsm1:value>
    </modification>
  </modifications>
</modifyRequest>
```

Account Containers

Hierarchical namespace accounts, such as ADS, LDAP, eWac, NDS, and PLS (eTrust SSO WAC Namespace), are stored in containers.

An account object in a hierarchical namespace is identified by the following ID string:

```
Account=xyzaccount,Container=ChildContainer,Container=ParentContainer,EndPoint=EX
AMPLE_ENDPOINT,Namespace=EXAMPLE_NAMESPACE,Domain=EXAMPLE_DOMAIN,Server=Server
```

If there is more than one container type in the endpoint, specify "Container.type=" instead of just "Container=" in the ID string. Container types are set to be the same as the LDAP objectClass of the container entry.

Example: Create an Account Container

The following example creates an account container on an ADS endpoint system of type ADSOrgUnit:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<addRequest xmlns="urn:oasis:names:tc:SPML:1:0">
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">

<id>Container.ADSOrgUnit=ADSSubContainer,Container.ADSOrgUnit=ADSContainer,EndPoi
nt=EXAMPLE_ADS_ENDPOINT,Namespace=ActiveDirectory,Domain=EXAMPLE_DOMAIN,Server=Se
rver</id>
  </identifier>
  <attributes/>
</addRequest>
```

Example: Create an Account within a Sub-Container

The following example creates an account on an ADS endpoint system within an existing sub-container:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<addRequest xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
  <operationalAttributes></operationalAttributes>
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">

<id>Account=EXAMPLE_ACCOUNT,Container.ADSOrgUnit=ADSSubContainer,Container.ADSOrg
Unit=ADSContainer,EndPoint=EXAMPLE_ADS_ENDPOINT,Namespace=ActiveDirectory,Domain=
EXAMPLE_DOMAIN,Server=Server </id>
  </identifier>
  <attributes>
    <attr name="objectClass">
      <dsml:value>user</dsml:value>
    </attr>
    <attr name="password">
      <dsml:value>test123</dsml:value>
    </attr>
    <attr name="NT_AccountID">
      <dsml:value>egaccount</dsml:value>
    </attr>
  </attributes>
</addRequest>
```

Complex Attributes

Objects, such as User, have attributes. Most attributes are of simple types like string, integer, or Boolean. The “address” or “createStatistics” attributes, however, are of complex types as they contain nested elements. For example “street,” “city,” “country,” “state” and “postcode” are nested fields of an “address” attribute. When you add a complex attribute, the Identifier of the complex attribute has the following special format:

attributeName@ID_Of_The_Actual_Object

Some complex attributes are multi-valued, such as the list of self authentication questions and answers for a Global User.

For multi-valued complex attributes, the Identifier format is as follows, with the #index to indicate the index of the attribute value. The index always start from 0:

attributeName#index@ID_Of_The_Actual_Object

Simple attributes can be populated when you add an object:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<addRequest xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsm1="urn:oasis:names:tc:DSML:2:0:core">
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>User=new_global_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
  </identifier>
  <attributes>
    <attr name="accountId">
      <dsm1:value>new_global_user</dsm1:value>
    </attr>
    <attr name="firstName">
      <dsm1:value>new_global_user</dsm1:value>
    </attr>
  </attributes>
</addRequest>
```

But complex attributes must be populated afterwards in a separate AddRequest.

Example: Add a Single-Valued Complex Attribute

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<addRequest xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsm1="urn:oasis:names:tc:DSML:2:0:core">
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>address@User=new_global_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
  </identifier>
  <attributes>
    <attr name="street">
      <dsm1:value>123 Church St </dsm1:value>
    </attr>
    <attr name="postcode">
      <dsm1:value>3121</dsm1:value>
    </attr>
  </attributes>
</addRequest>
```

Example: Add a Multivalued Complex Attribute

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<addRequest xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsm1="urn:oasis:names:tc:DSML:2:0:core">
  <operationalAttributes></operationalAttributes>
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>selfAuthQA#00@User=new_global_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
  </identifier>
  <attributes>
    <attr name="answer">
      <dsm1:value>Sample Answer</dsm1:value>
    </attr>
    <attr name="question">
      <dsm1:value>Sample Question</dsm1:value>
    </attr>
  </attributes>
</addRequest>
```

When you search for an object, asking about a complex attribute, the attribute value returned is a special attribute Identifier that refers to the real attribute value stored in a separate search result entry.

Request Retries

SPML requests, such as add, modify, delete and rename, can be flagged for retry. The request should be asynchronous and should be given a unique requestID. In addition, operational attribute calamRetry should be set to true.

For more information about operation retries, see the chapter "SPML Service."

Example: N16 Account-Add Request Flagged for Retry

```
<?xml version="1.0" encoding="UTF-8"?>
<addRequest execution="urn:oasis:names:tc:SPML:1:0#asynchronous"
requestID="AddN16Account" xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsm1="urn:oasis:names:tc:DSML:2:0:core">
  <operationalAttributes>
    <attr name="calamRetry">
      <dsm1:value>true</dsm1:value>
    </attr>
  </operationalAttributes>
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>Account=new_account,EndPoint=localhost,Namespace=Windows
      NT,Domain=EXAMPLE_DOMAIN,Server=Server</id>
  </identifier>
  <attributes>
    <attr name="password">
      <dsm1:value>myPassword</dsm1:value>
    </attr>
  </attributes>
</addRequest>
```

Propagate Global User Changes

Modifications made to global user attributes can be propagated to the global user's accounts, by setting the "syncAccounts" attribute to "true" in the modification request.

Example: Modify a Global User and Propagate Changes to Associated Accounts

```
<?xml version="1.0" encoding="UTF-8"?>
<modifyRequest xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
  <operationalAttributes></operationalAttributes>
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>User=_spml_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
  </identifier>
  <modifications>
    <modification name="comments" operation="replace">
      <dsml:value>new comment</dsml:value>
    </modification>
    <modification name="syncAccounts" operation="replace">
      <dsml:value>true</dsml:value>
    </modification>
  </modifications>
</modifyRequest>
```

But because the SPML Manager doesn't display "syncAccounts" attribute in the "Modify" tab for any object apart from the Global User. So, to propagate changes made to Global Users' complex attributes like "address", the user will have to manually add the operational attribute "syncAccounts" to the modification request. This is done by pressing the "Show Hidden Attributes" button then select "New Operational Attribute". Once the new field is added, specify the name of the field to be "syncAccounts" and set it to true.

Example: Modify Complex Attribute and Propagate Changes to Accounts

```
<?xml version="1.0" encoding="UTF-8"?>
<modifyRequest xmlns="urn:oasis:names:tc:SPML:1:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
  <operationalAttributes>
    <attr name="syncAccounts">
      <dsml:value>true</dsml:value>
    </attr>
  </operationalAttributes>
  <identifier type="urn:oasis:names:tc:SPML:1:0#DN">
    <id>address@User=_spml_user,Domain=EXAMPLE_DOMAIN,Server=Server</id>
  </identifier>
  <modifications>
    <modification name="city" operation="replace">
      <dsml:value>new city</dsml:value>
    </modification>
  </modifications>
</modifyRequest>
```

Escaping Special Characters in Object Identifiers

There are two special characters in the Provisioning Server object ID that need to be escaped when using SPML. If there is a comma character in the object name then you will have to use a backward slash to escape the comma. If there is a backward slash in the name then you have to escape it by another backward slash.

For example the identifier

"User=\\new\\,user,Domain=EXAMPLE_DOMAIN,Server=Server" identifies the user by the name "\\new,user" inside the domain EXAMPLE_DOMAIN.

Escaping Special Characters in Search Filters

If you need to search for a pattern that includes a special character *,), (, \ or NULL, it must be escaped using the format '\\code' (the code is actually the 2 hexadecimal characters representing the ASCII character) as follows:

- \\2a replaces or escapes *
- \\28 replaces or escapes (
- \\29 replaces or escapes)
- \\5c replaces or escapes \
- \\00 replaces or escapes NULL

Escaped Search Examples

(name=*\\2a*)	# searches for * anywhere in the name
(file=d:\\5cmyfile.html)	# searches for d:\\myfile
(description=*\\28*\\29)	# searches for both (and) anywhere and in that order
(bin=\\5b\\04)5b04	# searches for binary values (or unicode characters)

Appendix A: Provisioning Server Maintenance

This section contains the following topics:

[Back Up and Restore the eTrust Directory Database](#) (see page 183)

[Fine-tune the eTrust Directory Database](#) (see page 184)

[View and Maintain Log Files](#) (see page 185)

[Shut Down the Provisioning Server service](#) (see page 192)

Back Up and Restore the eTrust Directory Database

To ensure that data is coherent across your entire organization, regular backups should be done on all the domains at the same time. Regular backups of your eTrust Directory databases prevent data loss and damage caused by network disasters and failures. eTrust Directory provides the `dxbackupdb` and `dxrestoredb` utilities for you to back up and restore your eTrust Directory. For information about these utilities, see the chapter "Using DXtools" in the *eTrust Directory Administrator Guide*.

Important! Backups of your databases are extremely important and protect you against any event requiring you to restore the provisioning directory. For more information, see the chapter "Deployment" in the eTrust Directory Administrator Guide.

eTrust Directory recommends that three Directory servers (DSAs) be deployed for each database, e.g. three DSAs for the main provisioning database, another three for the notification database, and so on. Those DSAs should be configured multi-write for replication and with their respective databases.

To avoid down-time, one DSA from each database can be brought offline, leaving the other two DSAs serving the provisioning requests. The databases of those offline DSAs can then be backed up, which provides a consistent snapshot of the entire deployment. The two online DSAs can still offer real-time provisioning directory services, while database backup is being performed. After the backup, the offline DSAs can be brought back online again, allowing the other two DSAs to synchronize the updates occurred during the off-line period.

If there is only one Directory server deployed for each database, then you should stop the Provisioning Servers before you perform a backup.

Note: You should stop the Provisioning Server service before you perform a backup, so that the database cannot change during the backup process.

Fine-tune the eTrust Directory Database

If you experience performance problems, you can fine-tune your eTrust Directory database. This should be done after acquiring large directories or associating several objects. For information about the `dxtunedb` command, see the chapter "Using DXtools" in the eTrust Directory *Administrator Guide*.

The `dxtunedb` command can be run in two modes.

`dxtunedb {databaseName}`

- Only updates the Ingres statistics
- Can be run while the DSAs are online

`dxtunedb -full {databaseName}`

- Rebuilds DB indexes and updates the Ingres statistics
- Cannot be run while the DSAs are online.

Each deployment can decide how the databases need to be tuned. For example, an enterprise directory may decide to run daily online tunes, which only updates Ingres database statistics. This will improve the search performance, as it optimizes the search statistics used by the Ingres query analyzer. In addition, an offline tune can be performed once every month to rebuild the indexes.

To fine-tune the Provisioning Directory database:

1. Stop the Provisioning Directory Service.

```
dxtunedb -full etrustadmin
```

This command collects information, reorganizes data pages, and recreates indexes in the database.

2. Start the Provisioning Directory Service.

The notification database is used to synchronize the operations between Provisioning Server and Identity Manager and should be tuned regularly based on the synchronization workload as well as after heavy synchronization workload. Run the following command to tune the notification database:

```
dxtunedb -full etaops_notify
```

Performance Tuning with notSearchable Attributes

To provide full-scale search performance, most of the attributes in the provisioning directory are indexed. This requires additional index management overhead. For attributes that Provisioning Servers never search on, the not-searchable flag can be specified for further performance tuning.

The not-searchable attributes are specified in %DXHOME%\config\database\etrustadmin.dxc with a command such as:
set not-searchable = eTConfigParamDescription, eTConfigParamProperties, eTConfigParamValue, eTConfigParamOrigin, eTUPOExitPayload, eTUPOExitCustomData;

You may decide that additional attributes can be made non-searchable in your installation. Candidates for being made non-searchable are attributes that are set on many objects but are never used in search filters to locate objects.

Note: Attributes that are specified in the database configuration file also need to be created in the database itself. Use dxindexdb -notSearchable to create indexes in the database. If the provisioning directory is on a UNIX system, use etaindexdb.sh instead of using dxindexdb - notSearchable. The etaindexdb.sh is in \$DXHOME/samples/etrustadmin.

View and Maintain Log Files

The provisioning components (Provisioning Server, Connector Servers, Provisioning Manager) can be configured to log information about all transactions that they process. You can use this information to predict and identify the sources of system or security problems. For example, if the warning messages in log files show that some accounts on an endpoint could not be explored, you can use the logged information to investigate those accounts and determine why they were not explored. Use a text editor to view and edit provisioning log files.

Server Event logs track messages generated by the Provisioning Server. You can log messages to several optional destinations, including eTrust Audit.

The provisioning components provide other types of logging to diagnose specific problems. Other than the provisioning server trace log, these logs are usually not enabled unless you need them to trace a particular event. They include provisioning server logs, slapd logs, and C++ Connector Server logs. You can also diagnose problems that occur when communicating with the provisioning server by enabling Provisioning Manager logging.

Messages from all logs are written to text files in the *PSHOME\Logs* directory are named accordingly:

- Provisioning Server Event Log — *etayyyymmdd.log*
- Provisioning Server Trace Log — *etatransyyyyymmdd-hhmm.log*
- Provisioning Server IMS Notification Log — *etanotifyyyyyymmdd-hhmm.log*
- Provisioning Server SLAPD Log — *im_ps.log*
- Provisioning Manager Log — *etaclientyyyyymmdd.log*
- C++ Connector Server Endpoint Log — *sayyyymmdd.log*
- C++ Connector Server Trace Log — *satransyyyyymmdd-hhmm.log*
- C++ Connector Server SLAPD Log — *im_ccs.log*

Server Event Logging

Server Event logs record important events generated from the Provisioning Server. These events consist of all *severity levels* (success, information, warning, fatal, and error). The logs record every client-initiated operation and its success or failure, including generated sub-operations.

In the System Task frame of the Manager, under Global Properties, use the Logging tab to configure Server Event logging. Server Event logs typically only need to be configured once.

In some cases, you can turn logging on or off, or you can configure the severity levels of the messages logged. Thus, this Server Event logging can serve to audit the activities that are taking place within the Provisioning Server. However, the preferred auditing of provisioning activity is to enable the IMS Notifications features. The IMS Notifications feature sends detailed audit records to the IMS server for inclusion in the full audit record of Identity Manager activity. The notification records sent to the IMS can also trigger events for additional Identity Manager Server processing.

Endpoint Logging

In the Endpoint Task frame, you can configure endpoint-specific logging. Endpoint logs track messages that a connector generates when it processes requests for objects residing in that endpoint. Each endpoint can be configured separately so you can turn logging on or off for just the endpoints where you need to learn additional information to diagnose problems.

You can also specify the severity (success, information, warning, fatal, and error) of the messages that get logged.

To turn logging on or off and to set the logging destinations and the severity levels of the messages logged for each directory, use the Logging tab of the endpoint's property sheet in the Manager. For detailed instructions, see [Setting Endpoint Logging](#) in the Provisioning Manager help.

Endpoint logging is sent to a log file for the connector server in which the connector for the endpoint runs. For C++ connectors, the default log file name is *PSHOME\Logs\saYYYYMMDD.log*. The C++ connector server also adds some additional messages to this log. You control the log file name in the *im_ccs.conf* using the *BaseLogFileName* parameter. And you control which severities of these other messages are logged in the same conf file using the *LogSeverities* parameter.

Endpoint logging from connectors which run directly within the provisioning server (for example, the CA ACF/2 connector) log to the provisioning server's event log which has the default name of *PSHOME\Logs\etaYYYYMMDD.log*.

Diagnostic Logging

To diagnose specific problems, you can enable the provisioning server trace log, slapd logs, or C++ Connector Server logs. These are typically not enabled unless you need them to trace a specific type of event. Provisioning Manager logging also is used for diagnosing problems in the Provisioning Manager or client utilities.

Provisioning Server Trace Log

Enable this logging component to generate a special transaction log file that records the details of every transaction processed by the Provisioning Server. You can choose from several logging levels to match the level of logging detail you prefer using the domain configuration parameter *Transaction Log/Level*.

Note: For more information, see "[Advanced Configuration Options](#) (see page 201)."

The Provisioning Server trace log writes messages to *PSHOME\Logs\etaTransyyyymmdd-hhmm.log*. To change the base part of the file name (the part before the date) or to relocate this log file to another drive, modify the domain configuration parameter *Transaction Log/File name*. For more information about the *etaTransyyyymmdd.log* file, see the Provisioning Manager help.

Note: Unlike most logging which is turned off by default, Provisioning Server logging is fully enabled as the component is installed. If you choose not to run with maximum trace logging of the provisioning server, you need to change the domain configuration parameters that control this logging. These parameter are located in the "Transaction Log" parameter folder in the Provisioning Manager on the System task under Domain Configuration.

Provisioning Server IMS Notification Log

The Provisioning Server is typically configured to send notifications (global user and other object change records) to the Identity Manager Server for integration with the IMS event system and audit data base. A notification thread running within the Provisioning Server reads notification records from the local notify DB and transmits them to the IMS. This activity is captured in the IMS Notification log, whose name is *PSHOME\Logs\etanotifyyyyymmdd-hhmm.log*.

You configure the severity of log messages included in this log on the Identity Manager Setup screen in the System Task of Provisioning Manager.

The format of this log is similar to the Provisioning Server and Connector Server trace logs.

SLAPD and C++ Connector Server Logs

On Windows, you can enable SLAPD logging for advanced debugging tasks such as LDAP protocol packet handling and search-filter processing. You can set the log level in the Windows registry by assigning a value to the DebugLevel key. There are two registry keys, each controlling the logging for one of the services:

HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\slapd\eta_slapd\CurrentVersion\DebugLevel

The im_ps registry key controls logging for im_ps.exe, run by the Provisioning Server service.

HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\slapd\eta_connector\CurrentVersion

The im_ccs registry key controls logging for im_ccs.exe, run by the Identity Manager Provisioning Connector server service.

On Solaris, you enable SLAPD logging by setting the loglevel parameter in im_ps.conf or im_ccs.conf. Each file contains configuration instructions.

Note: While you may set the loglevel parameter in eta_slapd.conf and im_ccs.conf on Windows, that setting is ignored because the registry setting takes precedence. On Solaris, the registry setting is unused; you control the logging level by the configuration file value.

The DebugLevel registry key or loglevel configuration file parameter specifies the amount of information the server writes to its log file, which is one of the following, depending upon your type of slapd service:

- *PSHOME\Logs\im_ps.log*
- *PSHOME\Logs\im_ccs.log*

The debug level is set to the bit-wise OR of all the arguments on the configuration line. Each number is a decimal integer value. The debug level is taken as a bit string, with each bit corresponding to a different kind of trace information.

You can select a debug level to match the type of debugging you want to perform. The debug levels are listed in the following table:

Value	Debug Information
1	Trace function calls
2	Debug packet handling
4	Heavy trace debugging
8	Connection management
16	Print out packets sent and received
32	Search filter processing
64	Configuration file processing
128	Access control list processing
256	Stats log connections/operations/results
512	Stats log entries sent
1024	Print communication with shell back-ends
2048	Entry parsing
65535	All tracing

C++ Connector Server Trace Logging

C++ Connector Server Trace Logs record the activity of the C++ Connector Server, which is a module used to help manage many endpoint types. This log performs the following functions:

- Logs trace and debug messages for the C++ Connector Server.
- Monitors all statuses returned by the its connectors. For example, if a connector returns fatal LDAP errors, the C++ Connector Server logs these errors with severity LOG_FATAL.

To set the log file name and logging levels in `im_ccs.conf` set the `SATransLog` and `SATransLogLevel` parameters. The supported logging levels are 0 (for off) and 1 (for on). The default is 0. These parameters must exist in the file after the database superagent line.

Manager Logging

To diagnose problems communicating with the server, you can set logging to record events that transpire between the Provisioning Manager and the Provisioning Server to which it is connected. Use the Logging tab under File, Preferences to trace all requests sent to any server from the Provisioning Manager.

This logging is actually logging within the C/C++ client library used by the Provisioning Manager and some other clients (batch utility, password manager, csfconfig, bindeta, pingeta). Once logging is enabled and configured using Provisioning Manager, those log settings apply for these other clients as well. Each client logs its command name as it logs messages so you can identify which log messages are specific to which client.

However, for this to work the client being run must reside in same filesystem folder as the Provisioning Manager's etadmin.exe program. When this isn't the case (such as when running on Solaris where there is no Provisioning Manager install, or even on Windows when you run utilities from the Provisioning Server's installation), the client library consults registry settings specific to the provisioning server instead of specific to the provisioning manager. Set these other registry settings by running these eta-env commands using the eta-env program included in the provisioning server installation:

```
eta-env
  action=set
  name=Manager/LogMaster
  type=int
  value=1

eta-env
  action=set
  name=Manager/LogDestinations
  type=int
  value=16

eta-env
  action=set
  name=Manager/LogSevFile
  type=int
  value=31
```

These have the effect of configuring the C/C++ client library for the provisioning server's installation, setting the destination to "text file" and logging all message severities.

Finally, the csfconfig command has a "debug=yes" command-line parameter you can specify to turn this logging on for one command invocation overriding any registry settings configured with Provisioning Manager or eta-env.

Use AnalyzeLog

The command-line utility, AnalyzeLog, takes as input a provisioning server trace log (*etatransyyyymmdd-hhmm.log*) and produces different views of the information depending on what options you set. You can use this information to diagnose functional or performance problems reported by users.

For more details on this utility, see the Provisioning Manager online help.

Restore the Default Admin Profiles

If your default admin profiles are accidentally modified or deleted, Provisioning Manager lets you restore them. The predefined admin profiles are read-only and cannot be modified or deleted unless you are working in the provisioning directory directly.

To restore the default admin profiles, perform the following steps on the system where the provisioning directory is installed:

Step 1. Disable eTrust Directory Authentication

To disable eTrust Directory authentication, perform the following steps from a command prompt:

1. Change the working directory:

```
cd %DXHOME%\config\settings
```
2. Save a copy of the auth config file:

```
copy etrustadmin.dxc etrustadmin_auth.dxc
```
3. Create a new auth config file:

```
copy etrustadmin_anon.dxc etrustadmin.dxc
```
4. Instruct eTrust Directory to read the configuration files:

```
dxserver init etrustadmin
```

Step 2. Restore the Admin Profile

To restore the admin profile, perform the following steps from a command prompt:

1. Change directories:

```
cd PSHOME\bin
```
2. Restore the profile (press enter at the LDAP password prompt):

```
ldapmodify -c -x -D "" -W"" -H "ldap://localhost:20391" -f adminprof.ldf
```

Step 3. Enable eTrust Directory Authentication

To enable eTrust Directory authentication, perform the following steps:

1. Change directories:

```
cd %DXHOME%\config\settings
```

2. Restore the saved auth config file:

```
copy etrustadmin_auth.dxc etrustadmin.dxc
```

3. Issue the following command to instruct eTrust Directory dxserver to read the configuration changes:

```
dxserver init etrustadmin
```

Shut Down the Provisioning Server service

If you the Provisioning Server service will not shut down, you can manually shut it down as follows:

1. Open a command prompt and enter the following command:

```
net stop im_ps
```

2. If Services indicates that the Provisioning Server service is still in the stopping state, issue the following commands:

```
net start im_ps
```

```
net stop im_ps
```

A similar procedure can be used to manually shut down the Provisioning Connector Server service, whose service name is im-ccs.

If the service still does not stop, open the Task Manager, select im_ps.exe (or im_ccs.exe) on the Processes tab, and click End Process.

Appendix B: Provisioning Servers on UNIX

The Provisioning Server, the C++ Connector (SuperAgent) Server and various utilities that work with these servers can run on either Windows or UNIX platforms. For the most part the servers and utilities behave the same and therefore are documented with a single description. This appendix describes the major differences based on the operating system.

This section contains the following topics:

[No UNIX GUI Clients or Utilities](#) (see page 193)

[Command Line Examples](#) (see page 194)

[Libraries and Executables](#) (see page 194)

[Registry Access](#) (see page 195)

[Parser Tables](#) (see page 196)

[UNIX Services for Provisioning](#) (see page 196)

[Working with Hung or Crashed Servers](#) (see page 197)

[Scheduling Periodic Actions](#) (see page 197)

[Passwords on Command Lines](#) (see page 198)

[Server Event Logging Destinations](#) (see page 198)

[Program Exit Definitions](#) (see page 198)

No UNIX GUI Clients or Utilities

Other than installation and web clients, no graphical user interface (GUI) clients or utilities exist on UNIX. For example, the Manager (etadmin.exe) with its endpoint type-specific GUI plug-ins run only on a Windows system and access the UNIX Provisioning Server remotely. Also, the pwdmgr utility has a different format on UNIX, without a GUI, and the reporting clients must be run from Windows.

Command Line Examples

The Identity Manager documentation includes examples of invoking commands from a command prompt. On Windows, this prompt is a Command (DOS) Window; on UNIX, the prompt use one of various shells. Except where noted, you can assume the examples are for Windows. You can understand the environment variables and path separators that would be necessary to use a given commands on UNIX by replacing Windows pathnames such as

`%VARNAME%\data\im_ps.conf`

with

`$VARNAME/data/im_ps.conf`

Also, on UNIX nearly all directories (folders) and file names are in lower-case. Since case is significant in file names on UNIX but insignificant on Windows, some examples in the documentation that refer `PSHOME\Data` function correctly on Windows, even though new installations name that folder `data` instead of `Data`. If you are unsure about the case used for a directory on UNIX, use the `ls` command to locate the exact directory name.

When UNIX examples are given, they apply to any command shell, but were specifically tested to work with the Bourne shell (`/bin/sh`).

Also, note that quoting rules are different in Windows and UNIX command interpreters. Consult the respective interpreter documentation for how to quote or escape data that requires quoting or escaping.

Libraries and Executables

Libraries and executables differ on UNIX and Windows as follows:

Windows

Libraries are named `LibraryName.DLL` (mixed-case and `dll` suffix) and typically installed into a folder such as `PSHOME\bin`.

Executable programs are called `ProgramName.EXE` (mixed-case, `exe` suffix) and installed into `PSHOME\bin`.

Scripts are named `Script.bat` (mixed-case, `bat` suffix).

Message files are named `FileName.DLL` (mixed-case and `dll` suffix) and installed in `PSHOME\bin`.

UNIX

Libraries are named `liblibraryname.so` (lower-case, `lib` prefix and `so` suffix) and installed into a directory such as `$PSHOME/lib`.

Executable programs are called `programname` (lower-case, no suffix) and installed into `$PSHOME/bin`.

Scripts are named `script` or `script.sh` (lower-case, optional `sh` suffix).

Message catalogs are named `filename.res` (lower-case and `res` suffix) and installed in `PSHOME\data`.

Registry Access

On Windows, configuration information is stored in the Windows registry and edited with a native Windows utility such as `regedt32` or `regedit`. On UNIX, the registry is emulated as files in the file system (`/opt/CA/SharedComponents/eTrustCommonServices/registry`). Protect these files as you would the contents of other configuration files. Installation will protect the Identity Manager keys by default. Only `etadmin` group users can read them and only the `imps` user can write them.

To dump out the entire registry, you can use the command `eCSoption /r`. To view, modify or delete specific registry settings that are specific to Identity Manager, use the Identity Manager utility `eta-env`, which is documented in the Implementation Guide.

For example to view a registry setting, you could use these commands:

```
eta-env action=get name="etrust_bindtodb_need_tls" type=int
eta-env action=get name="logging/caldap_client_logfile"
eta-env action=get name="/enterprise_common_services/installpath"
```

and to set a registry value

```
eta-env action=set name="etrust_bindtodb_need_tls" value=1 type=int
eta-env action=set name="logging/caldap_client_logfile" value=my_file_name
```

Names that begin with `/` (slash)

```
[HKEY_LOCAL_SYSTEM]\SOFTWARE\ComputerAssociates
```

Simple names, without a `/` (slash), are relative to:

```
[HKEY_LOCAL_SYSTEM]\SOFTWARE\ComputerAssociates\Identity Manager\Provisioning
Server
```

Consequently, these two command invocations

```
eta-env action=get name="/Identity Manager\Provisioning
Manager/etrust_bindtodb_need_tls" type=int
eta-env action=get name="etrust_bindtodb_need_tls" type=int
```

refer the same configuration parameter.

Note: The registry path of “Identity Manager\Provisioning Server” is set in the \$ETAHOME/data/reg_path.conf file. The preceding eta-env commands are valid on UNIX and Windows; however, Windows has multiple eta-env.exe commands installed. If you run the eta-env.exe command from the provisioning server installation, it consults the reg_path.conf file from that installation and the registry keys and values are as shown in this section with UNIX. However, if you run the eta-env.exe command that is installed with the provisioning manager installation, it consults the reg_path.conf file from that installation and the registry keys and values being accessed are those under “Identity Manager\Provisioning Manager” instead.

Parser Tables

Parser table files are compiled files with suffix ptt that are installed in PSHOME\data on Windows (\$PSHOME/data on UNIX). They are read by the Provisioning Server and various utilities, such as dumpptt, etautil, showpttdit, and schemagen. The format is a platform-neutral format so that it can be freely copied between Windows and UNIX systems.

UNIX Services for Provisioning

The Provisioning Server (im_ps.exe) and C++ Connector Server (im_ccs.exe) are typically run as services on Windows. Thus you would typically start and stop them on Windows by going to the Services application. Alternatively you could start and stop them from the command line with commands such as net start im_ps and net stop im_ccs.

On UNIX, the Provisioning Server executable is called slapd and both servers normally start automatically through control files installed in /etc/rc*.d. To view, start, and stop the services manually, you can use commands such as imps status, imps start im_ps, and imps stop im_ccs. The command “imps” is also available as the command “eta” for backwards compatibility with prior eTrust Admin installations.

Working with Hung or Crashed Servers

On Windows, a crashed server may cause information to be written to the system's `drwtsn32.log` file, a file that CA Customer Support may ask you to send to help analyze the problem.

On UNIX, a crashed server creates a core file in `$PSHOME/bin` unless you have configured your server not to generate core files. If a core file is generated, please do not send it to CA unless instructed to do so. Instead, run the command `pstack core > pstack.txt` to capture the stack traces of all threads running within the crashed application. This output is valuable in diagnosing the failure.

On Windows, a hung server (one where one or more requests did not run to completion) can generally only be debugged using the provisioning server trace log (`$PSHOME\logs\etatranyyyyymmdd-hhmm.log`) in conjunction with the `analyzelog` utility. You will generally be asked to capture the provisioning server trace log (at logging level 7 if at all possible) and CA will use "analyzelog" to locate operations that have not yet completed. The C++ Connector Server trace log (`satransyyyymmdd-hhmm.log`) and sometimes other logs are also useful to collect.

On UNIX, capturing the provisioning server trace log and running `analyzelog` is still useful. But another option that often provides additional information is once again the `pstack` command. Locate the process ID (pid) of the hung service by reading the contents of the file `$PSHOME/data/pid/servicename.pid`, and then issue the command `pstack pid > pstack.txt` to capture the stack traces of all active threads within the running process. Please include this output file along with the provisioning server trace logs.

Scheduling Periodic Actions

The UNIX cron command is useful for scheduling periodic tasks such as script invocations. This includes invocations of `etacutil` commands (for checking or performing synchronization of accounts or users or performing refresh explore or correlate operations) and invocations of other utilities, such as `etadailybatch` and `etacreateouglocalgroups`. However, using `etacutil` for invoking period explore or correlate operations is no longer recommended. Instead you can configure explore/correlate tasks directly within Identity Manager.

Passwords on Command Lines

In UNIX, command-line arguments are public to anyone who can use the `ps` command on the UNIX system. Therefore, you should never supply a password or other sensitive information as a command-line argument. Each Identity Manager command accepts input from a file so you can avoid entering data on the command line. Often, the command-line parameter is still allowed for backwards compatibility with Windows.

Server Event Logging Destinations

Some of the server event logging destinations behave differently on UNIX from how they behave on Windows. In particular, the System log destination logs to `syslogd` on UNIX and to the Windows Event Viewer on Windows. Also, the eTrust Log destination should be avoided on UNIX. It logs to a local file that cannot be viewed locally since there is no UNIX version of the eCS Log Viewer utility. It is not recommended that you run the eCS Log Daemon on UNIX to export the log contents to a remote Windows system because you cannot control who can view the log remotely. These same logging destinations also apply to directory-level logging.

Program Exit Definitions

When defining a common program exit, you enter fields that are interpreted by the Provisioning Server, which invokes the program exit routine. In entering these fields, consider the operating system (Windows or UNIX) of that domain's Provisioning Servers so that these fields work on that operating system. If the domain includes Windows and UNIX Provisioning Servers, be sure that these fields work on both operating systems.

For SOAP program exits, the WSDL can be specified by a URI or its fully qualified name as seen from the Provisioning Server or by a pathname relative to `PSHOME\bin` (`$PSHOME/bin` on UNIX), which is the current working directory of the Provisioning Server.

For DLL program exits, the library name can be a fully qualified name or it can be a common name with or without the lib prefix or the .dll or .so suffix. When only one Provisioning Server exists for a domain, no restrictions exist for how you specify the library name. But when a domain has multiple servers, the library name must be valid for all the servers and since UNIX and Windows have different path syntaxes, files systems, prefixes, and suffixes, the library name should be defined as a common name without any prefix or suffix.

Thus the preferred way to define a program exit object for the CommonExit sample exit is to enter the string CommonExit for library name (in this exact case). The UNIX server will search LD_LIBRARY_PATH for a library named libCommonExit.so. On Windows, this will locate CommonExit.dll by searching the PATH environment variable.

Appendix C: Advanced Configuration Options

This section contains the following topics:

[Advanced Configuration Options Overview](#) (see page 201)

[Global Properties](#) (see page 203)

[Domain Configuration](#) (see page 204)

Advanced Configuration Options Overview

The advanced configuration options for the Provisioning Server fall into two categories:

Global Properties

Configuration settings that are saved in the provisioning directory and control the behavior of the Provisioning Server. Click the Global Properties button on the Provisioning Manager System Task frame.

Domain Configuration

Configuration settings that are saved in the provisioning directory and control the behavior of the Provisioning Server. Click the Domain Configuration button on the Provisioning Manager System Task frame.

There are additional places to configure various components of the Provisioning Server, including the following:

PSHOME\Data\im-ps.conf (on Windows)

Parameters control general behaviors of the Provisioning Server service not controllable through Domain Configuration parameters. See the Provisioning Manager help.

PSHOME\Data\im_ccs.conf

Parameters control general behaviors of the Identity Manager C++ Connector Server service. See the Provisioning Manager help.

PSHOME\PAM\etapam_id.conf

Parameters control an optional Pluggable Authentication Module you may choose to install. For more information, see the chapter Password Tools or the Provisioning Manager help.

PSAHOME\data\eta_pwdsync.conf

Parameters control an optional Password Synchronization Agent you may choose to install on a Windows system. For more information, see the chapter Passwords on Endpoint Accounts.

Note: *PSAHOME* is the directory where the Password Synchronization Agent is installed.

%DXHOME%\config\knowledge*.dxg and *.dxc

Used to configure eTrust Directory failover. For more information, see the *High Availability Guide*.

Global Properties

Global Properties are stored in the provisioning directory. These properties control the entire enterprise and as such are stored outside of any of your domain-specific data. To view these properties you must have a privilege that grants Read access to the SystemSettings object in the <ETA> domain. All predefined admin profiles in any domain grant this read access.

To change these properties you must have a privilege that grants Modify access to the SystemSettings object in the <ETA> domain. The DomainAdministrator and DomainAdministrator-NoWeb admin profiles in any domain grant this modify access.

Updates to global properties in most cases take effect immediately, no restarting of services or programs is necessary. Two specific exceptions are:

- Properties that control Manager behaviors, such as UID controls, and Full Name controls, do not affect property sheets that are already displayed. You may have to close a property sheet and reopen it to have the change take effect for your Manager.
- Logging settings are broadcast to affected Provisioning and Identity Manager Provisioning Connector server services. However, this broadcast currently only goes to one Provisioning Server service per domain. If you have installed a failover or load-balancing configuration with multiple Provisioning Server services for a single domain, you will need to restart all Provisioning and Connector Server services for that domain to ensure that the new logging settings are recognized by all affected components.

Domain Configuration

Domain configuration parameters are also stored in the provisioning directory. You manage these from the System Task of Provisioning Manager using the Domain Configuration button. Parameters are organized into a tree hierarchy using folders so that related parameters are easier to manage. These parameters control the Provisioning Server for a single domain.

If you configured multiple alternative servers, each with its own Provisioning Server for the same domain, all servers for the domain share the same configuration parameter settings. There are a few parameters that you might want to set to different values on different servers, even in the same domain. Per-server values are referred to as specializations. Use the Add Specialization or Remove Specialization menu items to work with server-specific values. These server-specific specializations are displayed in the tree hierarchy under the domain parameter. If there is no specialization for a particular server, the domain parameter value applies to that server. In most cases, the Provisioning Server lets you create a specialization for a parameter even if that could result in inconsistent behaviors from the alternative servers for a domain. This lets you have a dedicated server used for a specialized purpose where you actually want that different behavior. However, for a small set of parameters, specializations are not allowed. A typical reason would be because client code needs to know the value of the parameter even when it does not know which server handles its request. In those cases, the Add Specialization menu item is disabled.

To view these parameters, you must have a privilege that grants Read access to Configuration Parameter objects in the domain where they reside. All the predefined admin profiles grant this Read access to the domain configuration parameters in their own domain, including any subordinate domain.

To change these parameters, you must have a privilege that grants Modify access to Configuration Parameter objects in the domain where they reside. The DomainAdministrator and DomainAdministrator-NoWeb admin profiles grant Modify access to the domain configuration parameters in their own domain and any subordinate domain. You can create a custom admin profile that grants Read or Modify access to specific configuration parameters if you need scoping control.

Domain Configuration parameter updates take effect immediately on the provisioning server where the update was processed. However, if you configured multiple alternative provisioning servers for the domain, the other servers will not take the changed parameters into account immediately. The updated parameters are stored in the provisioning directory immediately, but each affected Provisioning Server refreshes its knowledge of the parameter values periodically. By default the update frequency is every 10 minutes; however you may change this value with the parameter Configuration Setup/Parameter Update Time described later. Thus you will need to wait up to 10 minutes for the refresh to take place. The refresh is recorded in the Provisioning Server Trace log with messages that include the text “ETA::Configuration update completed”.

Note: For more information, see the [Transaction Log](#) (see page 254) section.

You may choose to restart the affected Provisioning Server services to ensure the parameters are updated. When the service starts, the service writes information to the Provisioning Server trace log about configuration parameters. This log can be valuable in understanding what parameters were in effect at any particular point. The following information is written to the trace log at startup:

- If the Transaction Log/Level domain configuration parameter is set to a value of 0 or greater and Transaction Log/Enabled is Yes, non-default configuration parameters values are written.
- If the Transaction Log/Level domain configuration parameter is set to a value of 1 or greater and Transaction Log/Enabled is Yes, all configuration parameters values are written.

Note: A few parameters do not take effect even after the periodic configuration parameter update. They only take effect on restart of the Provisioning Server service. Such parameters display the following warning on their properties: Changing this parameter will require restarting all affected servers.

Provisioning Directory Parameters

The provisioning directory configuration folder contains parameters you can use if you have a non-default installation of your provisioning directory.

Provisioning Directory/Entry Count Attribute

Values: dxEntryCount (default) or <unset>

Description: Set this attribute to dxEntryCount if eTrust Directory is being used for the provisioning directory; but clear it otherwise. This attribute is used in queries sent to the provisioning directory to check whether size limits will be reached; but only if the client is not requesting partial results.

Authentication Parameters

The Authentication configuration folder contains parameters you can use to customize user authentication behaviors of the Provisioning Server.

Authentication/Disable Maintenance User

Values: No (default) or Yes

Description: Set this parameter to yes to disable the ability to authenticate to the Provisioning Server using the built-in user with the Distinguished Name `cn=etaserver,dc=eta`. This user, whose password is controlled by the `pwdmgr` utility, is used internally during installation.

After installation, this user is only needed for maintenance functions such as resetting an administrator's password. We recommend that you disable this user after installation.

Authorization Parameters

The Authorization configuration folder contains parameters you can use to customize authorization behaviors of the Provisioning Server.

Authorization/Check Owner Access on Indirect Privileges

Values: Yes (default) or No

Description: Controls what access checks are performed when assigning a global user group or admin profile to a global user, global user group or admin profile.

Regardless of the setting of this parameter, the Provisioning Server checks for Modify access to a specific attribute of the object being assigned and Modify access to a specific attribute of the object to which it is assigned.

If this parameter is Yes (the default), the Provisioning Server will also check for Owner access to each of the objects to which the assigned admin profile or global user group grants access. This prevents one from being able to assign privileges through a global user group or admin profile that one could not have assigned directly to the target global user, global user group or admin profile.

If you do not need added protection, this parameter can be set to No to disable additional Owner access checks. Doing so lets you have one set of administrators who define admin profiles and another set of administrators who assign those admin profiles to users.

Cache Parameters

The Cache configuration folder contains parameters that allow you to tune the Provisioning Server's use of its internal caches. Caches are used in the Provisioning Server to save information read from the provisioning directory so that it does not need to be read repeatedly in the same operation or across multiple operations.

Important! Changes to cache parameters do not take effect until the Provisioning Server service is restarted.

Each cache is controlled by the following parameters:

Maximum Age

The maximum time in seconds that an item remains in the cache without being reread from the provisioning directory.

Maximum Size

The maximum number of unused items to retain in the cache. While a cache item is being used by an operation, it is considered in-use, and there is no limit on the number of in-use cache items. However, when all operations finish with the cache item, it is marked unused and retained only when the number of used and unused items in the cache is no more than the configured maximum size.

Cache items are also removed from a cache when explicitly canceled. This occurs when a change is made to the provisioning directory data from which the cache item originates. This cache invalidation only occurs on the Provisioning Server that processed that provisioning directory update. If you have multiple provisioning domains or alternative servers serving a single domain, other servers may have cache items still derived from the prior data. That is why there is a cache maximum age parameter.

Cache items also are canceled when access is to be denied. The privilege caches (Admin Profile, Global User and Global User Group) contain privilege information used to perform authorization checks. If you have recently assigned a privilege to someone, you do not want to have to wait up to 10 minutes (the default cache maximum age for these caches) for that privilege addition to be recognized. Therefore if an authorization check using cached privileges is about to report DENIED, the cache items are canceled and re-initialized from the provisioning directory. If the result is still DENIED, that authorization failure is reported to the administrator.

Important! When you remove a privilege from a global user, admin profile, or global user group, expect that this change will take place at most 10 minutes (the default) from the time of the change. In most cases this is sufficient. However, if the reason for removing the access is to remove an imminent security threat, to ensure immediate enforcement of that privilege change requires you to restart all affected Provisioning Server services.

Admin Profile Privilege Cache

Each admin profile privilege cache item stores information obtained from an admin profile, including administrative privileges, and the names of included admin profiles.

Parameter: Cache/Admin Profile Privilege Cache Maximum Age

Default Value: 600 seconds (equals 10 minutes)

Parameter: Cache/Admin Profile Privilege Cache Maximum Size

Default Value: 10

Domain Cache

Each domain cache item stores information obtained from a Domain (DSA) Registration object. These objects record information that is necessary for one Provisioning Server to talk to another Provisioning Server.

Parameter: Cache/Domain Cache Maximum Age

Default Value: 3600 seconds (equals 1 hour)

Parameter: Cache/Domain Cache Maximum Size

Default Value: 20

Global User Group Privilege Cache

Each global user group privilege cache item stores information obtained from a global user group, including administrative privileges, the names of included admin profiles, and the names of included global user groups.

Parameter: Cache/Global User Group Privilege Cache
Maximum Age

Default Value: 600 seconds (equals 10 minutes)

Parameter: Cache/Global User Group Privilege Cache
Maximum Size

Default Value: 10

Global User Privilege Cache

Each global user privilege cache item stores information obtained from a global user and its assigned admin profiles and global user groups. This information includes administrative privileges, password, suspension status and the names of included admin profiles and including global user groups.

Unlike the admin profile privilege cache and global user group privilege cache, the global user privilege cache items also store indirect information obtained from referenced items, so it contains a full list of the accesses privileges that a global user has.

Each time a global user privilege cache item is initialized, the global user's full list of effective privileges and assigned admin profiles and global user groups is written to the server trace log. This information, written only if Transaction Log/Level is set to 4 or greater, includes information obtained directly from the global user, information obtained indirectly from assigned global user groups and admin profiles, and information obtained implicitly based on assigned web or workflow privileges. Look for the text "EFFECTIVE PRIVILEGE LIST INITIALIZED" in the server trace log. This will be followed by the distinguished name of a global user and a list of privileges, a list of admin profiles and a list of global user groups.

Parameter: Cache/Global User Privilege Cache Maximum Age

Default Value: 600 seconds (equals 10 minutes)

Parameter: Cache/Global User Privilege Cache Maximum Size

Default Value: 20

Notification Config Cache

The Notification Configuration Cache stores configuration information that drives the Identity Manager Server Notification feature. This configuration information, stored in the provisioning directory and updatable by the service utility `etoloadnotificationconf`, defines the mapping between provisioning actions and notification records that are sent to the IMS.

Parameter: Cache/Notification Config Cache Maximum Age

Default Value: 600 seconds (equals 10 minutes)

Parameter: Cache/ Notification Config Cache Maximum Size

Default Value: 10

Operation Cache

Each operation cache item stores information about an ongoing or recently completed operation. When you do an explore operation, for instance, the displayed Operation detail count value is obtained from an operation cache item.

Parameter: Cache/Operation Cache Maximum Age

Default Value: 600 seconds (equals 10 minutes)

Parameter: Cache/Operation Cache Maximum Size

Default Value: 10

Password Profile Cache

Each password profile cache item stores information from a password profile. Currently there is only one password profile per domain.

Parameter: Cache/Password Profile Cache Maximum Age

Default Value: 600 seconds (equals 10 minutes)

Parameter: Cache/Password Profile Cache Maximum Size

Default Value: 10

Connector Server Cache

Each C++ Connector Server cache item stores a pool of connections between the Provisioning Server and the C++ Connector Server. The C++ Connector Server server, also known as the Connector Server service, is the component that loads each endpoint type option's agent module.

Some endpoint types (for example Active Directory) provide a feature to use the administrator's own credentials rather than a configured set of proxy credentials for authenticating to the managed directory. Each C++ Connector Server cache item represents a pool of LDAP connections using a single set of administrator credentials.

Parameter: Cache/Connector Server Cache Maximum Age

Default Value: 3600 seconds (equals 1 hour)

Parameter: Cache/Connector Server Cache Maximum Size

Default Value: 20

Compatibility Parameters

The Compatibility configuration folder contains parameters you can use to provide temporary backwards compatibility with prior releases of eTrust Admin.

Values: No (default) or Yes

Description: By default (No), global user self authentication Q&A attributes are returned only when explicitly asked for by users. This allows the Provisioning Server to log when these questions and answers are viewed. Some older clients depend on the prior behavior where these attributes could be retrieved along with all other Global User attributes. Set this parameter to Yes to re-instate the prior behavior to allow these applications to work with the current Provisioning Server.

Configuration Setup Parameters

The Configuration Setup configuration folder contains parameters you can use to configure the processing of these domain configuration parameters.

Default Value: 600 seconds (equals 10 minutes)

Description: Configuration parameters are read periodically from the provisioning directory. This parameter defines how often, in seconds, this refresh of parameters occurs. Hence, this parameter defines the maximum amount of time one would need to wait after making a change to parameter before being assured the change has taken effect.

The minimum value for this parameter is 30 seconds.

Connections Parameters

The Connections configuration folder contains parameters you can use to tune the connection management mechanisms within the Provisioning Server.

The Provisioning Server maintains pools of LDAP connections that it uses for communicating with the provisioning directory, with connector servers and with other LDAP servers. A dedicated thread within the provisioning server (the connection monitor thread) wakes up periodically to adjust the pools by closing excess idle connections and attempting to create connections to LDAP servers previously believed to be unavailable.

The configuration parameters in this folder are consulted by the connection monitor thread as it performs its functions.

Connections/CS Pool Maximum Size

Default Value: 200

Description: The maximum size of each of the Provisioning Server's CS Connection Pools. A CS Connection Pool is a reusable set of LDAP connections that are used to communicate with a specific connector server.

Connections/CS Pool Minimum Size

Default Value: 2

Description: The minimum size of each of the Provisioning Server's CS Connection Pools. The connection monitor thread, when it closes expired idle connections, will retain at least this many connections in each CS Connection Pool.

Connections/DB Pool Maximum Size

Default Value: 40

Description: The maximum size of the Provisioning Server's DB Connection Pool. The DB Connection Pool is a reusable set of LDAP connections that are used to communicate with the Provisioning Directory.

Connections/DB Pool Minimum Size

Default Value: 5

Description: The minimum size of the Provisioning Server's DB Connection Pool. The connection monitor thread, when it closes expired idle connections, will retain at least this many connections in DB Connection Pool.

Connections/Expiration Time

Default Value: 1800 seconds (30 minutes)

Description: The time, in seconds, after which an idle connection in the provisioning server's LDAP connection pools will be considered expired. An expired connection is a candidate for being closed by the connection monitor thread.

Connections/Other Pool Maximum Size

Default Value: 20

Description: The maximum size of each of the Provisioning Server's Ad Hoc Connection Pools. Each Ad Hoc Connection Pool is a reusable set of LDAP connections that are used to communicate with a specific LDAP server other than the provisioning directory or regularly used connector servers. For example, changes to endpoint or endpoint type attributes may need to be sent to another provisioning server's connector server, and the connection pool to communicate with that connector server is governed by this parameter.

Connections/Other Pool Minimum Size

Default Value: 0

Description: The minimum size of each of the provisioning server's Ad Hoc Connection Pools. This value is typically zero as there is rarely a need to retain idle connections to these LDAP servers past their normal expiration time.

Connections/Refresh Time

Default Value: 300 seconds (5 minutes)

Description: The time, in seconds, that the provisioning server's connection monitor thread waits between iterations. Each time this thread awakens, it identifies expired connections in its LDAP connection pools and closes them. It also attempts to establish LDAP connections to servers that were believed to be unavailable (but only for pools with a minimum size greater than zero).

Endpoint Parameters

The endpoint configuration folder contains parameters you can use to enable or disable features on a endpoint type-by-endpoint type or endpoint-by-endpoint basis. Each parameter can be set to an ordered list of values, each of which can be one of the following:

Parameter	Description
ALL	Enabled for all endpoints of all endpoint types.
-ALL	Disabled for all endpoints of all endpoint types.
<i>EndpointType</i>	Enabled for all endpoints of the specified endpoint type.
<i>-EndpointType</i>	Disabled for all endpoint of the specified endpoint type.
<i>EndpointType:Endpoint</i>	Enabled for the specified endpoint.
<i>-EndpointType:Endpoint</i>	Disabled for the specified endpoint.

If more than one value for the same parameter specifies the same directory, the last value that specifies the endpoint determines whether the feature is enabled or disabled for that endpoint. This lets you provide a more general rule first (enabled for all directories or all endpoint types) and follow that up with a more specific rule (disabled for endpoint ABC of endpoint type ActiveDirectory).

Endpoint/Check Account Passwords

Default Value: -ALL (disabled for all endpoints of all endpoint types)

Description: When this parameter is enabled for a specific endpoint, the Provisioning Server checks any password in a password change of an existing account on that directory, including attempts to set an empty password.

During account creation, the Provisioning Server performs password quality checking when a password is provided. If no password is provided, no checking is performed unless the Check Empty Account Passwords parameter is also enabled for the directory.

Account password quality checking uses the Password Profile that exists in the domain of the global user that owns the account. If the account is not associated with any global user, then the Password Profile that exists in the domain of the account is used. If the password profile located based on the global user or the account's domain is disabled, account password quality checking is also disabled for that account.

Account password quality checking does not include the checks on self-changes that depend on history of recent password-change activity. Password reuse frequency (history) and minimum time between changes (interval checking) are only applicable to global user password changes where the Provisioning Server retains an accurate history of recent changes. Account passwords and password history are not stored in the Provisioning Server. They are stored only in the managed directory and the Provisioning Server makes no assumption that all password changes are visible to the Provisioning Server.

A synchronized account password is an account password meeting the following criteria:

- Account is correlated to a non-restricted global user
- Account resides on a directory for which Disable password propagation to accounts has not been enabled
- Account has not been deleted (it is not in Delete Pending state)

An attempt to change a synchronized account password to the value of the current global user password will be accepted regardless of the setting of this configuration parameter. Also, the settings of the following configuration parameters can control the effect of password quality checking on synchronized account passwords:

Passwords\Enforce Synchronized Account Passwords

Passwords\Use External Password Policies

Endpoint/Check Empty Account Passwords

Default Value: -ALL (disabled for all endpoints of all endpoint types)

Description: When this parameter is enabled for a specific managed endpoint, the Provisioning Server checks any empty password in an Add request for account on that endpoint. This parameter is ignored if Check Account Passwords is not also enabled for this endpoint.

This parameter is separate from Check Account Passwords because it is acceptable in some endpoint types to create an account with no password.

Endpoint/Use Account Template Status

Default Value: -ALL (disabled for all endpoint of all endpoint types)

Description: Parameter controls whether to ignore a global user status of suspended and allow the creation of accounts in the active state. By default accounts created from account templates for a suspended global user are suspended regardless of the suspended status indicated in the account template.

Endpoint/Validate Endpoint Credentials

Default Value: -ALL (disabled for all endpoint of all endpoint types)

Description: Parameter controls whether the Provisioning Server sends changes to passwords, updated on endpoint property sheets, to the applicable connector for immediate validation or if only the provisioning directory is updated. This functionality is not applicable to CA-provided connectors as they have all been updated to always have the behavior that is controlled by this parameter.

If you have a custom connector written using the Software Development Toolkit and that connector stores proxy credentials in its endpoint properties, verify the behavior of your endpoint type by enabling this parameter for your endpoints and attempting to change those proxy credentials.

Explore and Correlate Parameters

The Explore and Correlate configuration folder contains parameters you can use to configure the explore and correlate functions used while acquiring managed directories.

Explore and Correlate/Correlation Attribute

Value Syntax:

GUAttrName[=Namespace:AccountAttrName[:Offset,Length]]

Default Values:

GlobalUserName

FullName

Description: Controls the correlation matching algorithm used by the correlate phase of explore/correlate.

The correlation algorithm uses this parameter when determining how accounts are associated with global users.

Each value defines a global user attribute that will be compared against an account attribute. The list is ordered, and only values applicable for a endpoint type are used when correlating accounts from a directory of that endpoint type. If there are two defined mappings for the same global user attribute that are applicable to the endpoint type where correlate is being run, then the first parameter value is used.

You can provide this mapping in one of the following ways:

GUAttrName

In this form, you name only the global user attribute and not the corresponding account attribute. This value assumes for the omitted account attribute name the account attribute predefined by the endpoint type to correspond to this global user attribute. For information about the predefined mappings, see the endpoint type's *Connector Guide*.

A parameter value in this form applies to all endpoint types for which an account mapping is defined. All endpoint types define mappings for GlobalUserName (typically the account name). Most endpoint types define mappings for Full Name.

GUAttrName=Namespace:AccountAttrName

In this form, you name the global user attribute and a specific account attribute of a specific endpoint type. A parameter value in this form applies only to the indicated endpoint type. Use this form rather than the first form to match global users on an attribute such as Full Name in one endpoint type but not in all endpoint types.

GUAttrName=Namespace:AccountAttrName:Offset,Length

In this form, you name the global user attribute and a specified substring of an account attribute of a specific endpoint type. *Offset* indicates the start of the substring, the value 1 indicating the start of the attribute value. *Length* indicates the number of characters in the substring value. If the full account value is shorter than (*Length* + *Offset* - 1) characters, the substring value used will be shorter than *Length* characters.

A parameter value in this form applies only to the indicated endpoint type. Use this form if you know that an account attribute value (for example, description) has a form where the first 8 characters are known to contain a unique employee identifier that can be matched to a global user attribute value.

For example, assume the configured parameter values are the following:

```
GlobalUserName  
FullName=LDAP Namespace:globalFullName  
FullName=ActiveDirectory:DisplayName  
CustomField01=ActiveDirectory:Telephone
```

The following occurs for each previously uncorrelated account found while correlating accounts in an Active Directory container:

1. The Provisioning Server starts with the first parameter value (GlobalUserName) and determines that the Active Directory endpoint type's defined account attribute that maps to GlobalUserName is NT_AccountID (LDAP attribute name eTADSSAMAccountName). It attempts to find the unique global user whose name is equal to the account's NT_AccountID attribute value. If a unique match is found, the Provisioning Server associates the account with the global user. If more than one match is found, the Provisioning Server performs Step 5. If no match is found, the Provisioning Server performs the next step.
2. The Provisioning Server considers the second parameter value (FullName=LDAP Namespace:globalFullName). Since this value is specific to another endpoint type, it is skipped and the Provisioning Server performs the next step.
3. The Provisioning Server considers the third parameter value (FullName=ActiveDirectory:DisplayName). Since this value is specific to Active Directory, it is used. It attempts to find the unique global user whose FullName is equal to the account's DisplayName attribute value. If a unique match is found, the Provisioning Server associates the account with the global user. If more than one match is found, the Provisioning Server performs Step 5. If no match is found, the Provisioning Server performs step 4.

4. The Provisioning Server considers the final parameter value (CustomField01=ActiveDirectory:Telephone). Because this value is specific to Active Directory, it is used. It attempts to find the unique global user whose Custom Field #01 attribute is equal to the account's Telephone attribute value. Note that the name you gave to the custom global user attribute using global properties of the System Task is not displayed here. If a unique match is found, the Provisioning Server associates the account with the global user. If more than one match is found, the Provisioning Server performs Step 5. If no match is found, the Provisioning Server performs the next step.
5. The Provisioning Server associates the account with the [default user] object in the domain specified by the configuration parameter Explore and Correlate/Create Users Domain. If the [default user] object does not already exist, it is created.

Explore and Correlate/Correlation Domain

Values: Root Domain (default), Local Domain, or All Domains

Description: A value indicating which domain or domains should be searched for global users during the Correlate with existing global users phase of explore/correlate. If you place all global users into the root domain (or have only a single-domain installation), leave this parameter unchanged. However, if you distribute your global users into domains by geography or other organizational structure, select the value that reflects where the global users reside.

Note: Searching all domains is not efficient. If you have many domains and you know that most of the accounts in a directory or container will correlate to global users in one particular domain (root or local), you may be better off selecting that one domain. Any accounts not matching global users in that domain will be correlated to [default user]. If there are not many of those, you could manually correlate them to the correct global user by dropping each account onto the correct global user in another domain.

Explore and Correlate/Create Users Default Attributes

Value Syntax: *GUAttrName=Value.*

Default Values: None

Description: The parameter provides default values for global user attributes for global users created during the Create Global Users phase of explore/correlate.

For example, use 'SelfAdministration=1' to enable self administration for your new global users. Use this feature to assign constant values to optional global user attributes for global users created during the acquisition of a primary directory.

Explore and Correlate/Create Users Domain

Values: Root Domain (default) or Local Domain

Description: The domain in which global users are to be created during the Create Global Users phase of explore/correlate. If you place all global users into the root domain (or have only a single-domain installation), leave this parameter unchanged. However, if you distribute your global users into domains by geography or other organizational structure, select Local Domain to create the global users in the same domain as the primary directory you are acquiring.

The Create Users Domain also is used as the domain of the special global user [default user] used by Correlate to hold all accounts that could not be correlated to existing global users.

Explore and Correlate/Create Users Verify Not Correlated

Values: Yes (default) or No

Description: Temporarily set this parameter to No to enable the alternate behavior whereby the Create Global Users phase of explore/correlate will skip the check that the account is not already correlated to a global user.

When you acquire your primary directory or directories, you run the correlate function with the Create global users as needed option. The Create Global Users function works as follows for each account present in the container being correlated.

- Check to see if the account is already correlated to an existing global user. If so, leave this account still correlated to that global user. On an initial acquisition no accounts will be correlated. However, on a later re-explore/recorrelate this step is important so that the accounts remain correlated to the global user to which they were previously correlated.

Note: In a primary directory, you would not expect accounts to be correlated to any global user other than the one named the same as the account. There are various scenarios where this could occur. For instance, you may have renamed the account or the global user at some point after they were correlated to one another. Or you might have some system accounts on your primary directory that you do not want to correlate to separate global users - opting instead to correlate them to a single restricted global user.

- Attempt to create a global user named the same as the account. If this global user already exists, go on to the next step. This can happen if the global user was also present in another primary directory or you deleted your primary directory and re-acquired, correlating to global users created during the prior acquisition.
- Create an inclusion between the account and the global user, correlating the account with the global user.

If this configuration parameter is set to No, the first step is skipped. This can greatly improve the performance since that test is time-consuming and slows down as the provisioning directory becomes untuned. If you are acquiring a directory with more than thousands of accounts, the provisioning directory needs to be tuned, but you do not have the chance to do that tuning in the middle of the long-running Correlate operation.

Important! Set this parameter to No only during the initial acquisition of your primary directories. Subsequent use can result in accounts incorrectly correlated to multiple global users. Once the acquisition is completed, set the parameter to Yes.

Explore and Correlate/Map User ID to Lowercase

Values: No (default) or Yes

Description: Map all user IDs to lowercase when creating global users during the Create Users phase of explore/correlate. When you acquire one of your primary directories, you create global users for the accounts on that directory. Two of the global user properties that get set by this creation of global users are the following:

Account Name (LDAP attribute name eTUserId): Set to the corresponding account's account name property.

Global User Name (LDAP attribute name eTGlobalUserName): Set to be the same as the Account Name property, but translated to lowercase.

If you acquire a primary directory with mixed case account names, this will by default result in the created global user's Account Name property also having mixed case. Set the configuration parameter to Yes to force the Account Name property to be the same as the Global User Name property - always in lowercase.

Preserve the original case in the Account Name property by leaving the configuration parameter set to No if you have no endpoint types such as UNIX for which account names are case-sensitive.

An alternative to setting this parameter to Yes is to define your case-sensitive endpoint types' account template rule expressions for account name to use the TOLOWER built-in rule function, `$$$TOLOWER(%AC%)`, instead of the normal account name rule expression, which is `%AC%`.

Note: If your primary directory has only uppercase account names, this configuration parameter has no effect. The global user's Account Name property will already be translated to lowercase.

Explore and Correlate/Explore Compare in Memory

Values: No (default) or Yes

Description: Obtains two lists of objects at a time: one from the endpoint system being explored and one from the provisioning directory. These lists are compared in memory to determine what changes should be applied to the provisioning directory. If this parameter is no only a single list of objects at a time is explored and correlated, which uses far less virtual memory when working with large lists of objects.

Identity Manager Server Parameters

The Identity Manager Server configuration folder contains parameters you can use to control interactions between the Provisioning Server and the Identity Manager Server. Before enabling any of these parameters, you should configure the communication between the Provisioning Server and the Identity Manager using the “Identity Manager Setup” button on the System task of Provisioning Manager,

Identity Manager Server/Enable Corporate User Access

Values: No (default) or Yes

Description: Enables/disables retrieval of corporate user attributes from the Identity Manager Server during account template evaluation.

Important! This feature was not available at the publication time for this document. Please check the availability of the feature in the release notes before enabling this parameter.

Identity Manager Server/Enable Notification

Values: No (default) or Yes

Description: Enables/disables the collection of audit data (notifications) by the Provisioning Server (PS) for transmission to the Identity Manager Server (IMS). When enabled, any changes to data managed by the PS, other than changes directly initiated by the IMS, generate notifications which are queued in the Notify directory and then later sent to the IMS. Upon receipt at the IMS, certain notifications trigger events, while most are simply added to the full Identity Manager audit data.

Identity Manager Server/Notify Batch Size

Default Value: 100

Description: The number of notifications that are processed in one batch. When sending notifications to the Identity Manager Server, the Provisioning Server will retrieve at most this many records (a batch) from the Notify directory, process those entries, and then continue with additional batches.

Identity Manager Server/Notify Retry Time

Default Value: 600 seconds (10 minutes)

Description: The time, in seconds, that the notification thread pauses between iterations. The notification thread is a dedicated thread within the Provisioning Server that wakes up periodically and attempts to transmit (or retransmit) any queued notifications.

Identity Manager Server/Notify Timeout

Default Value: 30 seconds

Description: The timeout value, in seconds, for sending notifications or password validations to the Identity Manager Server. A value of zero indicates an unlimited timeout.

Identity Manager Server/Use External Password Policies

Description: When set to Yes, users changing their own global user passwords or one of their synchronized account passwords will have the password validated using externally-defined password rules. Users' synchronized account passwords are the passwords for their accounts on endpoints for which the Disable Password Propagation property is disabled. You should set the parameter Enforce Synchronized Account Passwords to Yes whenever Identity Manager Server/Use External Password Policies is set to Yes. When this parameter is set to Yes, the Provisioning Server password rules that are applicable to users changing their own passwords (Password history checks and Minimum interval between self-changes) are no longer consulted.

Values: No (default) or Yes

Note: Even when integration with Identity Manager password policies is enabled with this configuration parameter, the Provisioning Server uses its per-domain password profiles in various situations. In particular, Administrative password changes, initial global user passwords, changes to unsynchronized account passwords and generating temporary initial passwords all consult the Provisioning Server password profile. In addition, the Locking and Password Expiration features defined in the Provisioning Server password profile are always used. However, the Provisioning Server password profile rules that are applicable to users changing their own passwords (Password history checks and Minimum interval between self-changes) are not consulted when this configuration parameter is Yes.

Operation Details Parameters

The Operation Details configuration folder contains parameters you can use to control the behavior of operation details. Operation Details is the function that tracks the status of child operation spawned from higher-level operations such as Explore, Synchronization or Propagation. When you perform one of these higher-level operations from Identity Manager tasks or from Provisioning Manager, you receive a message in the message summarizing the results of the child operation.

The following is a sample summary message for a User Synchronization request:

(accounts created: 1, updated: 1, recreated: 0, failures: 0)

If you ask to view status details for the task (or double-click on the icon next to the summary message when using Provisioning Manager), this displays a screen with operation details including a series of success, failure, or warning messages corresponding to the statistics present in the summary message.

Operation Details/Maximum Operation Detail

Default Value: 100

Description: The maximum number of operation detail items which can be retrieved in one search of an operation object. When you perform a high-level operation that spawns hundreds or thousands of child operations and you call up the Operation Status window, this parameter controls how the details are returned from the Provisioning Server to the Provisioning Manager or other client application.

Operation Details/Operation Details Expiration Time

Default Value: 96 hours (equals 4 days)

Description: The number of hours to keep operation details in the provisioning directory.

Operation details are maintained in the server in the following parts:

1. An operation object stored in the provisioning directory (one per high-level operation).
2. An XML data file stored in the Operations folder containing the operation details, concatenated one after another.

Both objects are deleted when the operation object is deleted. Some clients delete their operation objects as soon as they retrieve the operation details or when the client terminates. Other clients such as Provisioning Manager leave the operation objects in the directory until they expire and are deleted in four days (by default).

Operation Details/Operations Folder

Default Value: Operations

Description: The name of the folder on the Provisioning Server where the XML data files storing operation details reside. This value can be a simple filename or a relative path name. However, it may not be an absolute path name.

Its value is relative to one of the following file path names:

%ETAVARHOME%
PSHOME
..

Normally, this means that the operations folder is placed alongside the Data and Logs folder with a path name like the following:

C:\Program Files\CA\Identity Manager\Provisioning
Server\Operations

However, to relocate this folder to another drive (so as to be able to run from a read-only drive), you should set the environment variable %ETAVARHOME% to a value such as D:\ProvisioningData before restarting the Provisioning Server service. Then the operations XML files will be placed instead into the following folder:

D:\ProvisioningData\Operations

The ETAVARHOME value can also be set as a registry value instead of an environment variable by using the eta-env utility that is installed with the provisioning server:

```
eta-env action=set name="ETAVARHOME" value="D:\ProvisioningData"
```

Important! Changes to this parameter do not take effect until the Provisioning Server service is restarted.

Password Synchronization Parameters

The Password Synchronization configuration folder contains parameters you can use to control the behavior of password synchronization operations. Password synchronization is the feature that involves installing the Password Synchronization Agent on a Windows system or other systems to intercept password changes, send password validation requests, and password notification requests to the Provisioning Server.

Password Synchronization/Agent Response Threshold

Default Value: 600 seconds (equals 10 minutes)

Description: Maximum expected duration (in seconds) of each password change that the Provisioning Server sends to a managed endpoint on which a password synchronization agent is installed. This parameter allows the Provisioning Server to recognize when a Password Synchronization agent is processing a password change sent to it by the Provisioning Server as distinct from a password change originating on that managed endpoint.

When installing a password synchronization agent, you must check that the Password synchronization agent is installed check box on the Endpoint Settings tab. Then when the Provisioning Server sends a password change to the managed endpoint, it records the time when the password was sent. For a number of seconds set by the Agent Response Threshold, any password change notification or password validation request received for this account is assumed to be false. Only password changes originating on the native system initiate password synchronization. Account password changes originating in the Provisioning Server update the account but not the global user or other accounts.

If, during the Agent Response Threshold, a password other than the password just sent to the managed endpoint is provided in a password validation or password change notification, this password is rejected. Two concurrent password changes to the same account are not allowed.

Password Synchronization/Update Only Global User

Values: No (default) or Yes

Description: This parameter controls what action is carried out when the Provisioning Server receives a password change notification. By default, the new Windows password received in a password change notification is used first to update the global user's password and then to update all of that global user's account passwords for accounts other than the one from which the notification arrived.

Set this parameter to Yes to change this behavior so that only the global user password is updated. No account passwords will then be updated.

There are various situations in which the global user and affected accounts are not updated, including the following:

- Global user Enable Password Synchronization Agent property is not enabled. Global Users and account passwords are not updated.
- Password change notification occurred during the Agent Response Threshold period and is treated as a false password change notification. Global Users and account passwords are not updated.
- A endpoint containing one of the global user's accounts is marked on its Endpoint Properties for Disable propagation to accounts. The accounts on this endpoint are not updated.
- Global user Restricted property is enabled. Restricted global users such as [default user] are protected from accidentally propagating changes to their associated accounts.

Password Parameters

The Password configuration folder contains parameters you can use to control the behavior of certain password operations.

Passwords/Enforce Synchronized Account Passwords

Values: No (default) or Yes

Description: When Yes, users cannot change any of their synchronized account passwords to a value other than the current value of their global user password. We recommend that you set this parameter to Yes whenever the Identity Manager Server\Use External Password Policies parameter is set to Yes.

Users' synchronized account passwords are the passwords for their accounts on directories for which the Disable Password Propagation property is disabled and which have not been marked as Delete Pending.

Passwords/Pre-expire Passwords

Values: No (default) or Yes

Description: Controls having new global users created with their passwords already expired, forcing users to change their passwords during the initial login. If you set this parameter to Yes, global users created from non-interactive interfaces have their password initially set as expired. This is represented in the global user properties as a value of 1 for the property PwdPreExpired. This option appears on the global user's property sheet as the Force one-time expiration (mark password as temporary) check box.

The setting of the password as initially expired occurs when global users are created through the following interfaces:

- Correlate. When acquiring a primary directory, global users are created for each account. These users will generally not have a password unless you set a constant value using the Create Users Default Attributes parameter described previously. Enabling the Pre-expire Passwords parameter will cause the global users to be created with passwords that are initially expired. If you set a value for PwdPreExpired using the Create Users Default Attributes parameter, that value takes precedence over one specified by enabling the Pre-expire Passwords parameter.
- Identity Manager Server, ETAUTIL or other on-demand clients. If you create a single global user using the batch utility (ETAUTIL) or some other on demand LDAP client, these users will start out with expired passwords if you enable the Pre-expire Passwords parameter and do not otherwise specify a value of the PwdPreExpired property.

If you create a global user using an interactive client such as Provisioning Manager. Then whether the global user's password is initially expired or not is determined from the value of the PwdPreExpired property provided when the global user is created. In Provisioning Manager, you control this value by selecting or clearing the check box labeled Force immediate expiration (one-time). Provisioning Manager automatically selects the Force one-time expiration (mark password as temporary) field if the Pre-expire passwords parameter is enabled. To disable this default behavior, clear the field before creating the global user.

Passwords/Store User Passwords

Values: Yes (default) or No

Description: Controls whether the EncryptedPassword global user attribute is stored and whether %P% rule variables are supported.

By default the Provisioning Server encrypts the global user password and stores it in the provisioning directory as a global user attribute named EncryptedPassword. When you later attempt to create an account for that global user using a policy with the %P% expression for the password rule, then the Provisioning Server decrypts the stored EncryptedPassword value and provides it to the endpoint type option as the initial Password attribute for the account being created.

However, if you will not be creating any accounts using account templates with %P% rule expressions, then you can improve security by not storing these passwords.

Note: By not storing the EncryptedPassword attribute, you are only giving up %P% rule evaluation. You can authenticate users by using the global user password. When the Store User Passwords parameter is set to No, the Provisioning Server stores a one-way hash of the password for use in authenticating user passwords during login.

Processes Parameters

The Processes configuration folder contains parameters you can use to control the process behaviors on Windows provisioning servers.

Processes/Catch Program Exit Exceptions

Default Values: Yes (default) and No

Description: This parameter controls the behavior of the Provisioning Server when invoked program exits throw runtime exceptions. By default (yes), the exception is caught and the current operation fails with an uncaught exception error message. However, if you are developing new program exits you may choose to set this parameter to no and allow the uncaught exception to result in server termination, which provides more information about the exception. This parameter only affects common program exits of the DLL type.

Processes/Child Operation Thread Pool Size

Default Value: 200

Description: This parameter defines the maximum number of threads in the server-wide child operation thread pool. When the server decides to split up a single operation into multiple sub-operations, those sub-operations are carried out by the threads in the child operation thread pool. The larger you make the value for this parameter, the more work the Provisioning Server attempts in parallel.

Currently the Provisioning Server only uses the child operation thread pool to carry out the multi-account search and multi-account update functions submitted by the Web interface. For synchronization and propagation operation, regardless of which client submits these requests, and even though they also spawn child operations, the child operations are carried out in series in the main operation's thread.

This parameter does not affect the primary server thread pool used for processing separate requests received from client applications. This thread pool size is controlled by the SLAPD parameter called `threads` in the `eta_slapd.conf` file. See on the Provisioning Manager help for editing parameters in this configuration file.

Important! Changes to this parameter do not take effect until the Provisioning Server service is restarted.

Processes/Parallel Propagation

Default Values: Yes (default) and No

Description: This parameter controls whether account passwords updated as part of global user to account password propagation are carried out in parallel or sequentially. By default, account passwords are updated in parallel, and the degree or parallelization is controlled by the Processes/Child Operation Thread Pool Size parameter.

This parameter has no effect on requests from clients that carry out the account updates explicitly. It only affects those clients that direct the Provisioning Server to update a global user password and propagate that change immediately to all synchronized account passwords.

Processor Parameters

The Processes configuration folder contains parameters you can use to control values related to the use of CPU processors. These parameters control operating system values that are process-wide and as such affect the entire Provisioning Server service. This is specifically relevant if you install any additional backends into the slapd process that runs your Provisioning Server.

Important! Changes to these parameters do not take effect until the Provisioning Server service is restarted.

Processor/Process Affinity Mask

Default Value: 0 (no restrictions)

Description: Specifies the process affinity mask for the Provisioning Server service process. The process affinity mask is a bit vector in which each bit represents the processor of a multiprocessor server on which the threads of the process are allowed to run.

The value 0 (default) signifies no restrictions.

The values 1, 2, 4, 8, 16, 32, 64, or 128 restrict the threads to running on the 1st, 2nd, 3rd, 4th, 5th, 6th, 7th, or 8th processor, respectively.

For example, the values can be combined so the value 5 (1+4) can be used to allow running on processors 1 and 3.

Processor/Process Priority

Default Value: 0 (use system default priority)

Description: Specifies the scheduling priority of the Provisioning Server service process.

The value 0 (default) uses the system default.

The only other recommended value is 16384, indicating below-normal priority. This value should be used when the Provisioning Server runs on the same server as its provisioning directory. This effectively raises the priority of the provisioning directory and consequently increases over-all server performance.

Search Parameters

The Search configuration folder contains parameters you can use to control search behaviors.

Search/Allow Partial Results

Values: Yes (default) or No

Description: Allow search requests to return less than the full number of matched entries when the search size limit is reached. If you set this parameter to No, or if the client fails to request partial results, partial results are not returned and clients receive a size limit exceeded error when the size limit is reached.

For example, if partial results are permitted, the search limit is 200, and the search found 5000 entries, 200 of them are returned. If partial results are not permitted, this search would return *no* results.

Note: The Provisioning Server does not define *which* 200 of the 5000 entries are returned. The Provisioning Server does not necessarily return the *first* 200 entries, either alphabetically or using any other ordering method.

Two settings are required to activate Partial Results:

- The domain configuration parameter Allow Partial Results described here.
- A selection in the Provisioning Manager GUI Search Preference control.

The effective partial result setting is the combination of these two settings. A partial result is returned only if Allow Partial Results is set to Yes *and* the GUI Search Preference control has Show Partial Result Lists selected.

The Provisioning Server is most efficient when partial results are not returned. When partial results are not needed, the Provisioning Server can report quickly when a search would exceed the size limit. However, if required to return the number of entries indicated by size limit, this will add processing load to the provisioning directory, Provisioning Server and client application. This load will take away from the processing load available to other users' queries or modifications.

Search/Max Scope Filter Objects

Default Value: 10

Description: The maximum number of objects that will be placed into a search filter during scoped searches. When a search of a container is initiated by an administrator who has access to only some of the objects in the container, the Provisioning Server augments the client-supplied filter with a scope filter that restricts the objects returned to those for which the administrator has access. However, if the administrator has access to more than Max Scope Filter Objects objects, this is deemed too many to be placed into the filter that will be sent to the provisioning directory and/or managed endpoint and the Provisioning Server uses its backup algorithm. In this case, the Provisioning Server will ask for all objects the client requested and then discard those that the scope filter would have excluded.

Search/Search Size Limit

Default Value: 0 (unlimited)

Description: The maximum number of entries returned by the Provisioning Server in a search request. The effective size limit for a search is the smallest of this value, the SLAPD size limit parameter, and the client-provided size limit operation parameter.

Limiting the number of entries returned in searches is important for good interactive performance of the Provisioning Server. If the effective size limit is too large, poorly formed search requests may return very long lists of global users, accounts or other objects that are not easy for administrators to work with. Conversely, if the effective size limit is too small, it may limit the administrator's ability to browse provisioning roles or other objects whose number is more moderate than that of global users or accounts.

There are three ways to control the effective size limit of a search request:

- The SLAPD *sizelimit* parameter in the following files:

PSHOME\Data\im_ps.conf

PSHOME\Data\im_ccs.conf

This value is set to 0 (unlimited) by default and controls all LDAP servers running in the Provisioning and Provisioning Connector server services, respectively. There should be no need to change this parameter. Doing so would limit operations like exploration that perform searches of accounts in managed directories and relies on being able to receive all accounts present in a single container in a single search request.

- The Domain Configuration parameter Search Size Limit

This value is also set to 0 (unlimited) by default. It controls the maximum number of entries returned to provisioning clients in a single search request. If you set this to a non-zero value (500), this will prevent any client from being able to receive more than 500 entries from the Provisioning Server in a search request.

Set this parameter only if you can be sure that no clients require receiving more than this number of entries from any search. Be aware that the Reporting client often requires receiving many items from the server. You may have additional non-interactive clients that similarly require receiving large search results.

You should leave the Search Size Limit configuration parameter as 0 (unlimited) on any interactive or mixed-use Provisioning Server. If you have an environment where certain Provisioning Servers are dedicated to interactive use and other Provisioning Servers are available for batch activity such as the generation of reports, you may want to set the Search Size Limit between 500 and 1000 on the interactive servers only. Use the Add Specialization menu item to set a server-specific Domain Configuration parameter.

- The Manager's size limit preference setting

To change this preference, select File, Preferences, click the Search tab, and change the value in the Limit on Returned Items for a Search field.

The Provisioning Manager preset return limit is 500. Each user can increase or decrease this preference, but increasing the value above the server's Search Size Limit value has no effect because the server's effective size limit is the smallest of the three size limit controls.

If a search operation encounters a search-limit failure, assuming that you enabled the retrieval of partial results, in some cases the number of items displayed may be different than the actual search limit because a single search operation (from the perspective of the user) might require several searches (from the perspective of the Provisioning Server).

Depending on how a client combines the results of the multiple search operations (for example, through a union or intersection) when displaying the results, the net display may contain more or fewer items than the search-limit. In all cases, an error message is displayed informing you that the results were truncated due to the search limits.

Servers Parameters

The Servers configuration folder contains read-only parameters that identify which Provisioning Servers are installed for your domain. After a default installation, a single server is listed.

If you install alternative servers for failover or load-balancing, multiple servers appear in this list. For each server listed, read-only parameters identify the Build, Patch and Version numbers for the Provisioning Server software. An additional parameter identifies whether the FIPS 140-2 encryption feature is enabled for that Provisioning Server.

Note: The server names listed here are the same server names used when creating specialization parameters. These are also the names you should use for the server parameter in the csfconfig command-line utility when creating specializations for connector server configuration objects.

Statistics Parameters

The Statistics configuration folder contains parameters you can use to control how statistics are maintained by the Provisioning Server. Most objects stored in the provisioning directory have statistic attributes to record when and by whom the object was created; and when and by whom the object was last updated. These statistics are displayed on the Statistics tab of the respective objects' properties.

Statistics/Enabled

Values: Yes (default) or No

Description: When disabled, statistic attributes on objects in the provisioning directory are not updated by the Provisioning Server as those objects are created or updated. This can improve performance during large scale changes or in installations where maintaining creation and update statistics is not necessary.

Certain statistics on global users such as password update date and time, and suspension update date and time are required for correct operation of server functions. These statistics are updated even when the Statistics/Enabled configuration parameter is set to No.

Statistics/Node Stats from Connection

Values: No (default) or Yes

Description: Use the client node name taken from the LDAP connection object when recording node statistics. The default (No) behavior is to take the node name provided by the client application.

The Node statistic displayed on the Statistics tab of the Global User property page and other objects' property pages is not always updated when other statistics such as date, time, userid, and username are. This behavior is a result of the way the Node name is determined. By default, the node name must be provided by client applications in their requests. If the clients fail to do so, or the client submits a high-level operation, such as Synchronize, that spawns child operations to carry out the individual object updates, the server has no Node value to use to update that statistic. To rectify this problem, use the Node Stats From Connection configuration parameter. Change its setting from the default No to Yes to select the alternative Node statistic algorithm.

The alternate algorithm uses the Node information obtained from the LDAP connection, which identifies the host that was the immediate client sending the request. This is often the same as the originating client, but can be another system.

For example, requests originating from the Identity Manager Server would be recorded as originating from the computer where the Identity Manager Server is running, not the computer where the administrator is running a web browser. Also, if the clients connect to a dXRouter process for failover or load-balancing between replicated Provisioning Servers and have the dXRouter send the request to the Provisioning Server, you should not enable the Node Stats from Connection parameter. It will result in all Node statistics indicating the router system.

Synchronization Parameters

The Synchronization configuration folder contains parameters you can use to choose from alternative variants of the Provisioning Server's synchronization functions.

Synchronization/Automatic Correlation

Values: No (default) or Yes

Description: Enable the alternative User Synchronization behavior whereby an attempt to update an existing, uncorrelated account triggers an automatic correlation of the account to the global user prior to the update of the account. If the parameter is No (default), the attempt to update the account will fail with a message indicating the account has not yet been correlated to this global user.

Synchronization/Remove Account Template Values from Accounts

Values: Yes (default) or No

Description: When Yes, the Weak Synchronization algorithm will consider that capability account values (for example, account group membership) prescribed by an account template should be removed when that account template is removed from an account. Set this parameter to No to restore the prior Weak Synchronization behavior where account attribute values are never removed when synchronizing an account with its weak-synchronization account templates. This parameter only affects multivalued attributes. String, integer or Boolean single-valued attributes are only increased in capability by weak synchronization.

Only certain multivalued attributes designated as SyncRemoveValues attributes are affected by this feature. Consult the eTaCapability.txt file for a list of which multivalued capability attributes may have values removed by the SyncRemoveValues feature described here.

To generate the eTaCapability.txt file, use the following command:

```
PSHOME\bin\dumpptt -c > eTaCapability.txt
```

Synchronization/Use Existing Accounts

Values: No (default) or Yes

Description: Enable the alternative User Synchronization behavior whereby a global user's set of assigned account templates (through assigned provisioning roles) will only attempt to prescribe one account correlated to the global user on any particular managed directory. This behavior can be useful if some accounts already correlated to the global user are named differently or are in different containers than what is prescribed by the account templates included in the global user's provisioning roles and only one account is needed or allowed. If the parameter is enabled and multiple account templates for one directory prescribe different names and/or different containers for the account only one account will be created.

If a global user already has multiple accounts on a single directory, the User Synchronization function (when Use Existing Accounts is set to Yes) attempts to figure out which account is required by which policy. This is done through a heuristic that attempts to handle situations where a user's provisioning roles do in fact prescribe multiple accounts on one directory.

For example, if global users have two accounts (A1 and A2) on endpoint E and their provisioning roles indicate that they should have one account on endpoint E through account template AT1 and one account on endpoint E through account template AT2, User Synchronization pairs each account template (AT1 and AT2) with one of the existing accounts. The pairing is done with the following heuristic:

- Match account template with an account with exactly the DN specified by the account template.
- Match account template with an account already belonging to the specified account template. If more than one account matches, pick the first one.

- Match account template with an account whose endpoint type-specific account name attribute matches the global user's name. In some endpoint types, for instance Active Directory, the account name is represented by an attribute of the account whereas the name as seen when you list the account is a display name (a full name). This rule accounts for such endpoint types.
- Account with name value matching the name value specified in the account template. That is, it matches an account with the right name but the wrong container. If there is more than one matching account, pick the first one.
- Pick the first account.

Note: When Use Existing Accounts parameter is set to No, only the first of these rules (exact matching based on account DN) is applied.

Continuing with the example, if the previous rules resulted in pairing both policy AT1 and policy AT2 with account A1, then User Synchronization would correct the accounts for this user by doing the following:

Deleting account A2 (assuming the administrator selected the Delete extra accounts or extra policy assignments option of User Synchronization); and

Assigning either account template AT1 or AT2 to account A1 that was not already assigned.

These rules ensure that User Synchronization (with Use Existing Accounts enabled) never attempts to create additional accounts on an endpoint where a user already has an account. If your business requires you to create multiple accounts for your users on a single endpoint from provisioning roles, do not enable this configuration parameter. For more information about synchronization, see Account Synchronization in the chapter "User Synchronization."

Transaction Log Parameters

The Transaction Log configuration folder contains parameters you can use to control transaction logging, also known as Provisioning Server trace logging. This is the log you use to monitor activity performed by the Provisioning Server while it processes requests received from its client applications.

Transaction Log/Enable

Values: Yes (default) or No

Description: Set this parameter to No to completely disable the logging of information to the server trace log. Typically, you control the amount of information you want logged using the Level parameter. However, even at level 0 some important items are logged to the server trace log. To disable these items from being logged, set the Enable parameter to No.

Transaction Log/Enable/Configuration

Values: Yes (default) or No

Description: This parameter enables or disables logging of diagnostic output from the Provisioning Server Configuration subsystem. The configuration subsystem checks every 10 minutes (by default) to see whether any of the configuration parameters have been changed. Each time this periodic refresh occurs, a line such as the following is written to the server trace log: ETA::Configuration update completed. No changes found.

Alternative messages are written if actual changes were found. To suppress all of these messages from the server trace log, set this configuration parameter to No.

Transaction Log/Enable/Connector Server Framework

Values: Yes (default) or No

Description: Enables/disables logging of the diagnostic output from the Provisioning Connector Server Framework (CSF).

Transaction Log/Enable/LDAP

Values: Yes (default) or No

Description: This parameter enables or disables logging of diagnostic output from the Provisioning Server LDAP subsystem. The LDAP subsystem manages the communications between each Provisioning Server and other LDAP servers, including the provisioning directory, the Java and C++ Connector Servers and Provisioning Servers that manage other domains.

Transaction Log/File Name

Default Value: etatrans

Description: This parameter defines the transaction log's base file name. The suffix *YYYYMMDD-HHMM.log* will be appended to this base file name to build the log file name. You can change this parameter to use a different base file name in the *PSHOME\Logs* folder or to relocate the log file to another folder on your server.

The value of this parameter can be any of the following:

- Simple base name (for example, etatrans), the log is created in the Logs folder in the folder where you installed the Provisioning Server. By default, this makes the log file named `C:\Program Files\CA\Identity Manager\Provisioning Server\Logs\etatransYYYYMMDD-HHMM.log`.
- Relative path (for example, `..\Logs\etatrans`), the log path name will be relative to the current directory of the Provisioning Server service (`PSHOME\bin`). For the example given, this will result in the same pathname as before (`C:\Program Files\CA\Identity Manager\Provisioning Server\Logs\etatransYYYYMMDD-HHMM.log`).
- Absolute path (for example, `D:\ProvisioningData\Logs\etatrans`), you can specify an alternative drive for your log file. For this example, the resulting log file would be `D:\ProvisioningData\Logs\etatransYYYYMMDD-HHMM.log`.

The Provisioning Server switches to a new log file every day, every time the Provisioning Server restarts, and any time the log file size exceeds 100 Megabytes.

Transaction Log/Level

Values: 0 through 7 (default)

Description: This parameter lets you set the level of logging for the Server Trace log. Valid values are:

Value	Description
0	No trans logging
1	Log external/child errors
2	Log external operations
3	Log child operations
4	Log informative messages
5	Log DSA (Directory Service Agent) errors

Value	Description
6	Log DSA operations
7	Log search operations

Note: Alert log entries are logged at all logging levels (1 - 7).

After installation, the log level is set to the maximum value (7). This ensures that any problems during or immediately after installation are logged. After installation, you may select alternative logging levels to meet your logging requirements. Many customers run with level 7 for maximum information in the event that problems are reported by users. Other customers select a more modest level such as level 3 that reports failures without much of the internal tracing information associated with the processing of requests. Another useful level is level 6 that removes the many search operations that could dominate the log while maintaining all other information.

Appendix D: Program Exits

This section contains the following topics:

[Program Exits Overview](#) (see page 259)

[Ordering of Program Exit Invocations](#) (see page 260)

[Basic Structure of Program Exits](#) (see page 262)

[Define Common Exits in the Provisioning Manager](#) (see page 262)

Program Exits Overview

Program exits let you write software that executes during certain Provisioning Server actions. Program exits let you reference custom code from in the Provisioning Server process flow, extending the framework of the Provisioning Server to allow additional functionality that changes or augments standard behavior. Numerous exit points are available where custom code can be referenced, depending on the type of object. For example, you may want to install some files on a system every time a UNIX account is created. You could write a program exit that performs the file creations, and specify that it be run whenever a UNIX account is created.

There are two types of program exits:

- *Common Exits* are executed from the Provisioning Server core infrastructure.
- *Native Exits* are executed from the managed endpoint types.

The type of program exit is determined by where it is handled, not where it is referenced.

Note: For information about writing common exits, see the *Programming Guide for Provisioning*. For information about native exits, see the endpoint type-specific *Connector Guide*.

Program exits are implemented as separate objects in the Provisioning Server namespace and are referenced in these objects, permitting you to define the necessary exits and associate them at the points where they need to be referenced. The following objects reference program exits:

- Common Configuration Objects
- Provisioning Roles
- Account Templates
- Endpoints

Each of these objects can reference multiple program exits, including multiple exits of the same type. For example, a directory can reference two exits that handle routines to be executed before creating an account.

Ordering of Program Exit Invocations

A single request processed by the Provisioning Server may make multiple program exit invocations. The order in which these program exits are invoked depends on:

- The type of program exit
- The location of the program exit reference
- The priority number assigned to the program exit reference

Each program exit type identifies a place in the Provisioning Server's control flow where that particular type of program exits gets a chance to affect the Provisioning Server's behavior. Therefore, to understand the order in which different program exit types are invoked requires understanding how requests are processed.

For instance, a single request to the Provisioning Server might change a global user password and then propagate that password to one or more of that user's accounts. The processing of this request is done as a high-level global user operation that spawns separate account operations.

This would result in invoking program exits in the following order:

1. PRE_CHANGE_GLOBAL_USER_PWD
2. PRE_CHANGE_ACCOUNT_PASSWORD
3. POST_CHANGE_ACCOUNT_PASSWORD
4. PRE_CHANGE_ACCOUNT_PASSWORD
5. POST_CHANGE_ACCOUNT_PASSWORD
6. POST_CHANGE_GLOBAL_USER_PWD

In some cases, multiple program exit types apply to the same object class (PRE_CHANGE_GLOBAL_USER_PWD and PRE_MODIFY_GLOBAL_USER) and could potentially be applicable to the same request (a single global user modification that changes both the password and full name, say). In such a case, all exits of one of these exit types will be called before all exits of another of these exit types. But the order is unspecified and you shouldn't assume that the ordering will remain unchanged in future versions of the Provisioning Server.

For a single exit type, sometimes you have a choice as to the class of object on which you define the program exit reference. In particular, references to some exit types that affect global users can be defined on a provisioning role (affecting only users in that role) or on the common configuration object (affecting all users in the domain). If you define exit references on both kinds of objects, then the Provisioning Server invokes the ones defined on the provisioning roles before invoking the ones that are defined on the common configuration object.

Similarly, references to some exit types that affect accounts can be defined on an account template (affecting only accounts assigned to that account template) or on the endpoint (affecting all accounts in the endpoint). If you define exit references on both kinds of objects, then the Provisioning Server invokes the ones defined on the account templates before invoking the ones that are defined on the endpoint.

Finally, exit references of the same type and defined on the same class of object are invoked in priority order using the priority number you assigned when you created the program exit reference, such as priority 1 first, priority 2 second, and so on. Two program exit references of the same priority are invoked in unspecified order.

Basic Structure of Program Exits

Program exits are referred to in terms of pre-operation or post-operation (that is, some operation that Provisioning Manager is used to performing). Program exits have a single common interface for their calling structure. This interface consists of a single argument and a single return value. The single argument is an XML buffer representation, encoded in Unicode Transformation Format 8 (UTF-8), of the object being acted upon combined with any custom information from the definition of the exit. The return value is status information about the result of the program exit execution as well as any documented custom information that is required for a particular exit.

Define Common Exits in the Provisioning Manager

You can define a common exit that will be executed in the Provisioning Server core infrastructure by using the Program Exit property sheet.

Note: To define a Native Exit to be executed in a managed endpoint type, see the specific *Connector Guide*.

To define a common exit, perform the following steps:

1. Click Endpoints.
2. Select Common Program Exit from the drop-down list in the Object Type field.
3. Click New. The Common Program Exit dialog appears.
4. Fill in the name and description of the exit to be invoked on the Program Exit tab. If the Disabled box is selected, the program exit will not be invoked, even if it is referenced in another object.
5. Specify whether the Provisioning Server uses the Simple Object Access Protocol (SOAP) or a Dynamic Link Library (DLL) file to invoke the exit on the Common Parameters tab. You can specify that the information be sent securely by selecting SSL Enabled.
6. Enter the path that points to the DLL file or the address of the SOAP service in the Location field. In the Method field, provide the name of the exported function in the DLL file or the name of the function defined by the SOAP service.

For DLL program exits, you can enter for location either a full path, such as:

`c:\yourfolder\yourlibrary.dll`

or you can enter just the common name of the library

`yourlibrary`

If you enter just the common name, you can have more than one Provisioning Server for the domain, where the library does not have to appear at exactly the same path. This is important if the domain has a mix of Solaris and Windows servers, because these operating systems have different pathname syntax.

If you provide just a common name (yourlibrary), the Provisioning Server will locate the library in the following way:

- For Windows, locate yourlibrary.dll file on the Provisioning Server service's execution path as defined by the PATH environment variable. We recommend that you place the library in the PSHOME\bin folder which is known to already be on PATH.
- For Solaris, locate the libyourlibrary.so file on the Provisioning Server service's library path as defined by the LD_LIBRARY_PATH environment variable. We recommend that you place the library into the \$PSHOME/lib directory which is known to already be on LD_LIBRARY_PATH.

7. Select an Authentication Type to provide information for authentication data to be passed to the invoked program exit on the Authentication tab:

- Select None to pass no authentication data to the exit.
- Select Current User to pass the authentication data of the global user who is logged on at the time the exit is invoked.
- Select Proxy User to select a specific global user that will be used for authorization of the operations.

Note: When you select Proxy User, the information you provide must be that of a valid global user.

- Select Other to enable the Authentication Details group field, which lets you select an arbitrary name and password. The exit code uses this information for authentication.

8. Click OK to complete the definition of the common exit.

Index

A

- Account Containers • 176
- Account Management Tasks • 41
- Account Synchronization • 108
- Account Template Management • 59
- Account Templates • 47
- Account Templates (Provisioning Policies) • 20
- Account Templates Overview • 47
- Account-Level Password Quality Checking • 79
- Accounts on Endpoint Systems • 14
- Accounts with Multiple Templates • 62
- Activate Syslog Messages • 96
- Activate System PAM Debug Mode • 87
- Activate Trace Messages • 97
- Add a New SPML Service • 134
- Add Account Templates to Accounts • 105
- Add Request • 162
- Add the Attribute Element to Identity Manager • 40
- Add Workflow Support for
 - AccumulatedProvisioningRolesEvent • 71
- Admin Profile Privilege Cache • 210
- Admin Profiles • 19
- Administrator Authentication • 18
- Administrator Authorization • 18
- Administrator Login • 18
- Advanced Account Operations • 43
- Advanced Configuration Options • 201
- Advanced Configuration Options Overview • 201
- Advanced Rule Expressions • 51
- Assign Orphan Accounts • 42
- Assign System Accounts • 43
- Attributes Only for New Accounts • 64
- Authentication Parameters • 207
- Authentication/Disable Maintenance User • 207
- Authorization Parameters • 207
- Authorization/Check Owner Access on Indirect Privileges • 207

B

- Back Up and Restore the eTrust Directory Database • 183
- Basic Structure of Program Exits • 262

- Batch Request • 162
- Benefits of Using SPML • 125
- Built-in Rule Functions • 56

C

- C++ Connector Server Trace Logging • 189
- CA Product References • iii
- Cache Parameters • 208
- Cancel Request • 164
- Capability and Initial Attributes • 48
- Change an account Template • 65
- Change the Global User for an Account • 44
- Check Account Synchronization • 109
- Check User Synchronization • 107
- Choices for Password Management • 75
- Class 'classname' is not a valid class name • 122
- CMDRA Command Options • 139
- CMDRA Commands • 139
- CMDRA Examples • 141
- Combined User Store and Provisioning Directory • 16
- Combining Rule Strings and Values • 52
- Command Line Examples • 194
- Command Line Requesting Authority (CMDRA) • 128
- Common Error Messages • 121
- Compatibility Parameters • 214
- Complex Attributes • 178
- Configuration • 88
- Configuration Setup Parameters • 215
- Configure Alternate Servers for the Password Synchronization Agent • 78
- Configure Password Synchronization • 81
- Configure Retry for a Request • 158
- Configure SPML Client Computer to Support SSL Security • 138
- Configure SSL Support for Tomcat Servers • 136
- Configure Synchronization in the Provisioning Manager • 29
- Configure the Inbound Administrator • 26
- Configure the pam.conf File • 91
- Configure the pam_CA_eta.conf File • 88
- Connect an Environment to the Provisioning Server • 28

- Connections Parameters • 215
- Connections/CS Pool Maximum Size • 216
- Connections/CS Pool Minimum Size • 216
- Connections/DB Pool Maximum Size • 216
- Connections/DB Pool Minimum Size • 216
- Connections/Expiration Time • 217
- Connections/Other Pool Maximum Size • 217
- Connections/Other Pool Minimum Size • 217
- Connections/Refresh Time • 218
- Connector Server Cache • 214
- Connector Server Framework (Super Agent Framework) • 21
- Connector Servers (Agent Plug-ins) • 20
- Contact CA • iii
- Could not find keyword xxxxx for class classname • 122
- Create a New JIAM Jar File • 33
- Create a Provisioning Role • 67
- Create Accounts • 104
- Create an SPML Template Request • 148
- Create Global Users from Identity Manager • 99

D

- Data Transformations • 150
- Default Account Template for an Endpoint Type • 51
- Default Admin Profiles • 19
- Define Common Exits in the Provisioning Manager • 262
- Define the JIAM Cache TTL (Time-to-live) • 73
- Delete Accounts • 45, 105
- Delete an Existing Service • 135
- Delete Global Users using Identity Manager • 101
- Delete Request • 164
- Deployment on JBoss • 38
- Deployment on WebLogic or WebSphere • 38
- Diagnostic Logging • 187
- Domain Cache • 210
- Domain Configuration • 204
- DOS Output from etautil • 123
- Download the SPML Manager • 147

E

- Enable Log Messages • 81
- Enable Password Synchronization • 101
- Enable Provisioning • 23
- Enable Provisioning Role Accumulation • 70

- Enable the JIAM Cache • 73
- End of file reached while expecting an operator • 121
- Endpoint Account to Global User Synchronization • 102
- Endpoint Accounts • 41
- Endpoint Logging • 186
- Endpoint Parameters • 218
- Endpoint Types (Namespaces) • 20
- Endpoint/Check Account Passwords • 219
- Endpoint/Check Empty Account Passwords • 221
- Endpoint/Use Account Template Status • 221
- Endpoint/Validate Endpoint Credentials • 222
- Endpoints (Directories) • 20
- Escaping Special Characters in Object Identifiers • 182
- Escaping Special Characters in Search Filters • 182
- etautil and Account Templates • 64
- etautil Batch Utility • 111
- etautil Control Statements • 114
- etautil Syntax • 112
- Example
 - Add a Multivalued Complex Attribute • 179
 - Add a Single-Valued Complex Attribute • 179
 - Create an Account Container • 177
 - Create an Account within a Sub-Container • 177
 - Modify a Global User and Propagate Changes to Associated Accounts • 181
 - Modify Attributes in Global Settings • 176
 - Modify Complex Attribute and Propagate Changes to Accounts • 181
 - Search for Attributes Defined in Global Settings • 175
- Example Data Transformation • 150
- Example of a Batch Request • 163
- Example of a Cancel Request • 164
- Example of a Delete Request • 165
- Example of a Modify Request • 169
- Example of a Modify/Propagate Request • 171
- Example of a Schema Request • 172
- Example of a Search Request • 173
- Example of a Status Request • 174
- Example of an Add Request • 162
- Example of an Extended Request • 165
- Explicit Global User Attribute Rules • 55

- Explore and Correlate an Endpoint • 32
- Explore and Correlate Parameters • 222
- Explore and Correlate/Correlation Attribute • 223
- Explore and Correlate/Correlation Domain • 226
- Explore and Correlate/Create Users Default Attributes • 227
- Explore and Correlate/Create Users Domain • 227
- Explore and Correlate/Create Users Verify Not Correlated • 228
- Explore and Correlate/Explore Compare in Memory • 230
- Explore and Correlate/Map User ID to Lowercase • 229
- Extended LDAP JIAM Jar • 36
- Extended Request • 165
- Extended Request Types • 166
- Extending Custom Attributes on Endpoints • 33

F

- Failover • 81
- Fields in a Delete Request • 164
- Fields in a Modify Request • 169
- Fields in a Search Request • 173
- Fields in an Add Request • 162
- Fields in an Extended Request • 165
- Fine-tune the eTrust Directory Database • 184
- Flow of the SPML Feed Command • 146

G

- Generating JAVA Source Files • 34
- Global Properties • 203
- Global Settings • 174
- Global User Group Privilege Cache • 211
- Global User Privilege Cache • 211
- Global User Synchronization • 104
- Global Users • 14

H

- How Automatic Exploration Works • 44
- How Capability Attributes are Synchronized • 60
- How PAM Works • 83
- How the Password Synchronization Agent Works • 79
- How the SPML Service Works • 129
- How to Acquire Endpoints • 31

- How to Configure an Environment for Provisioning • 26
- How to Define JIAM Custom Connector Extensions • 37
- How to Deploy JIAM Custom Connector Jar Files • 38
- How to Validate Custom Connectors • 38
- How UNIX PAM Works • 93

I

- IAMServer Initialization • 74
- Identity Manager Server Parameters • 230
- Identity Manager Server/Enable Corporate User Access • 231
- Identity Manager Server/Enable Notification • 231
- Identity Manager Server/Notify Batch Size • 231
- Identity Manager Server/Notify Retry Time • 232
- Identity Manager Server/Notify Timeout • 232
- Identity Manager Server/Use External Password Policies • 232
- Identity Manager Users • 13
- Import a Provisioning Role • 68
- Import Custom Provisioning Roles • 26
- Inbound Synchronization • 99
- Initial Configuration • 23
- Install SPML • 132
- Install the Password Synchronization Agent • 76
- Installation • 94
- Installation Directory • 21

J

- JIAM Object Cache • 73

K

- Key Provisioning Terms • 20

L

- Libraries and Executables • 194
- List Templating Variables • 149
- Locate Non-Compliant Accounts • 65
- Log Endpoint Messages • 33
- Log On to the SPML Configuration Application • 133

M

- Managed Endpoints • 31
- Manager Logging • 190
- Modify an Existing Service • 134
- Modify Request • 169
- Multiple Accounts for Users • 13
- Multivalued Attributes • 119
- Multivalued Rule Expressions • 53

N

- No UNIX GUI Clients or Utilities • 193
- Notification Config Cache • 212

O

- Object 'XXXX' operation failed
 - DB operation failed
 - Target DN not found. • 121
 - No server plug-in found for operation • 121
- Obtain Operation Details • 122
- Operation Cache • 213
- Operation Details Parameters • 233
- Operation Details/Maximum Operation Detail • 234
- Operation Details/Operation Details Expiration Time • 234
- Operation Details/Operations Folder • 235
- Order Provisioning Role Event Processing • 69
- Ordering of Program Exit Invocations • 260
- Outbound Synchronization • 99

P

- Parser Tables • 196
- Password Parameters • 237
- Password Profile Cache • 213
- Password Quality Enforcement • 81
- Password Synchronization • 75
- Password Synchronization Parameters • 236
- Password Synchronization/Agent Response Threshold • 236
- Password Synchronization/Update Only Global User • 237
- Passwords in Endpoint Accounts • 75
- Passwords on Command Lines • 198
- Passwords/Enforce Synchronized Account Passwords • 238
- Passwords/Pre-expire Passwords • 238
- Passwords/Store User Passwords • 240

- Performance Tuning with notSearchable Attributes • 185
- Pluggable Authentication Module (PAM) • 82
- Prerequisites • 93
- Processes Parameters • 240
- Processes/Catch Program Exit Exceptions • 241
- Processes/Child Operation Thread Pool Size • 241
- Processes/Parallel Propagation • 242
- Processor Parameters • 242
- Processor/Process Affinity Mask • 243
- Processor/Process Priority • 243
- Program Exit Definitions • 198
- Program Exit Rule Functions • 59
- Program Exits • 259
- Program Exits Overview • 259
- Propagate Global User Changes • 170, 180
- Propagate Initial Attributes • 60
- Provisioning Directory Parameters • 206
- Provisioning Directory/Entry Count Attribute • 206
- Provisioning Manager • 17
- Provisioning Overview • 13
- Provisioning Role Performance • 72
- Provisioning Roles • 67
- Provisioning Roles for Existing Environments • 74
- Provisioning Roles Management • 67
- Provisioning Server • 17
- Provisioning Server IMS Notification Log • 188
- Provisioning Server Maintenance • 183
- Provisioning Server Trace Log • 187
- Provisioning Servers on UNIX • 193

R

- RDUtility.bat Syntax • 36
- Recompute Template Rules • 64
- Recreate Deleted Accounts • 46
- Register an endpoint • 31
- Register the New JIAM Jar • 36
- Registry Access • 195
- Removing Account Templates from Accounts • 105
- Rename an Existing Service • 135
- Request Execution Types • 161
- Request Retries • 180
- Request Types • 162
- Requesting Authorities • 131
- Requirements • 94

- Restore the Default Admin Profiles • 191
- Retry Architecture • 155
- Retry Configuration Files • 156
- Retrying SPML Requests • 154
- Rule Substrings • 52
- Run the Installation Program • 77

S

- Sample PAM Programs • 83
- Sample SPML Requests • 161
- Scheduling Periodic Actions • 197
- Schema Request • 171
- Search Filters • 173
- Search Parameters • 243
- Search Request • 172
- Search/Allow Partial Results • 244
- Search/Max Scope Filter Objects • 245
- Search/Search Size Limit • 245
- Separate User Store and Provisioning Directories • 15
- Server Event Logging • 186
- Server Event Logging Destinations • 198
- Servers Parameters • 248
- Session Pooling • 74
- Shut Down the Provisioning Server service • 192
- SLAPD and C++ Connector Server Logs • 188
- SPML Architectural Diagram • 127
- SPML Architecture • 126
- SPML Configuration Application • 128
- SPML Feed • 142
- SPML Feed Command Options • 144
- SPML Integration • 130
- SPML Manager • 128
- SPML Overview • 125
- SPML Service • 125, 127
- SPML Service Configuration • 132
- SPML Templates • 130
- Statistics Parameters • 248
- Statistics/Enabled • 249
- Statistics/Node Stats from Connection • 249
- Status Request • 174
- Step 1. Disable eTrust Directory Authentication • 191
- Step 2. Restore the Admin Profile • 191
- Step 3. Enable eTrust Directory Authentication • 192
- Strong Synchronization • 61
- Synchronization Parameters • 250

- Synchronization/Automatic Correlation • 250
- Synchronization/Remove Account Template Values from Accounts • 251
- Synchronization/Use Existing Accounts • 252
- Synchronize Accounts • 109
- Synchronize Accounts with Account Templates • 65
- Synchronize Global Users or Roles • 107
- Synchronize Users in Create or Modify User Tasks • 107

T

- Tasks You Can Perform • 111
- Test the Configuration File • 97
- Test the LDAP/TLS Connection • 97
- Test the PAM Framework • 98
- Test the Password Synchronization • 98
- The JIAM Jar File • 34
- Transaction Log Parameters • 254
- Transaction Log/Enable • 254
- Transaction Log/Enable/Configuration • 254
- Transaction Log/Enable/Connector Server Framework • 254
- Transaction Log/Enable/LDAP • 255
- Transaction Log/File Name • 255
- Transaction Log/Level • 256
- Troubleshooting PAM • 86
- Troubleshooting UNIX Password Synchronization • 96

U

- Uninstall the SPML Service • 132
- UNIX Password Synchronization and PAM • 93
- UNIX Services for Provisioning • 196
- Unknown error nnn opening Common Object Repository • 121
- Update Global User Passwords • 85
- Update Global Users using Identity Manager • 100
- Use AnalyzeLog • 191
- Use Delete Pending • 45
- Use DeletePending • 120
- Use PAM • 84
- Use PAM with Multiple ADS Domains • 86
- Use Values Instead of Rule Strings • 51
- User Store and Provisioning Directory • 15
- User Synchronization • 99
- User Synchronization between Servers • 99
- Using PAM and Password Expiration • 84

Using the SPML Manager's Templating
Functionality • 147
Using Velocity Templates • 149

V

Values and Rule Strings • 49
Verify the Installation • 82
View and Maintain Log Files • 185
View and Modify Endpoint Accounts • 41

W

Weak Synchronization • 61
When You Would Use the SPML Service • 126
Why Global Users Become Out of Sync • 103
Working with Hung or Crashed Servers • 197
WS Mapper • 131