



Sun Java System Identity Synchronization for Windows 6.0 Deployment Planning Guide



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

Part No: 819-4635
March 2007

Copyright 2007 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, docs.sun.com, AnswerBook, AnswerBook2, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and SunTM Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

U.S. Government Rights – Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2007 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées du système Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, docs.sun.com, AnswerBook, AnswerBook2, et Solaris sont des marques de fabrique ou des marques déposées, de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

Contents

- Preface13**

- 1 Before You Begin Deployment 19**
 - Features of Identity Synchronization for Windows 19
 - Pre-deployment Requirements and Guidelines 20

- 2 Case Study: Deploying in a Multimaster Replication Environment 23**
 - Example Bank Deployment Information 23
 - Example Bank’s Existing Architecture 23
 - Example Bank’s Technical Requirements 25
 - Identity Synchronization for Windows Features in This Case Study 28
 - Deploying the Solution 28
 - Creating a Special Active Directory User for Identity Synchronization for Windows 29
 - ▼ To Assign Administration Rights to the Special User 29
 - Configuring the Identity Synchronization for Windows Core 30
 - Configuring Directory Sources 30
 - ▼ To Specify Information in the Global Catalog and for the Active Directory Domain . 32
 - ▼ To Specify the Windows NT Domain 35
 - Configuring the Synchronization Settings 36
 - ▼ To Configure the Creation Attributes 42
 - Configuring the Synchronization User Lists 43
 - Installing the Connectors and Directory Server Plug-Ins 51
 - Running idsync resync52
 - ▼ To Synchronize Attribute Values in Active Directory With the Values in Directory Server After Linking Entries 59
 - Configuration and Installation Summary 59
 - Migrating Users From Windows NT to Active Directory 61

3 Case Study: Deploying in a High-Availability Environment Over a WAN Using SSL	69
Global Telco Deployment Information	69
Global Telco's Existing Architecture	70
Global Telco's Technical Requirements	72
Identity Synchronization for Windows Features in This Case Study	74
Installation and Configuration Overview	74
Primary and Failover Installations	74
Periodically Linking New Users	80
Large Deployment Considerations	80
Configuration Walkthrough	80
Primary Installation	81
Failover Installation	88
Setting Up SSL	91
Increasing Connector Worker Threads	97
Aligning Primary and Failover Configurations	98
Setting Multiple Passwords for uid=PSWConnector	99
Initial idsync resync Operations	100
Initial idsync resync Operation for Primary Installation	101
Initial idsync resync Operation for Failover Installation	103
Periodic idsync resync Operations	104
Periodic idsync resync Operation for Primary Installation	104
Periodic idsync resync Operation for Failover Installation	108
Configuring Identity Manager	109
Understanding the Failover Process	109
Directory Server Connector	110
Active Directory Connector	110
Initializing the Connector State	111
Failover Installation Maintenance	112
When to Fail over	112
Failing Over	113
Stopping Synchronization at the Primary Installation	113
Starting Synchronization at the Failover Installation	114
Re-enabling the Directory Server Plug-Ins	114
Changing the PDC FSMO Role Owner	115
Monitoring the Logs	115
Failing Back to the Primary Installation	116

A	Pluggable Authentication Modules	117
	Advantages of Combining PAM and Identity Synchronization for Windows	117
	Configuring PAM and Identity Synchronization for Windows	119
	To Configure an LDAP Repository for PAM	119
	To Install and Configuring Identity Synchronization for Windows	122
	To Populate the LDAP Repository	123
	To Configure a Solaris System to Use PAM	126
	To Verify That PAM Is Interoperating With the LDAP Store	131
	To Demonstrate That User Changes Flow to the Reciprocal Environment	132
	▼ To Verify Entries on Windows	134
	Configuring Systems to Prevent Eavesdropping	137
	Introducing Windows NT Into the Configuration	137
	Example /etc/pam.conf File	138
B	Configuring Identity Manager and Identity Synchronization for Windows to Coexist	141
	Components for Deploying Identity Manager and Identity Synchronization for Windows ...	141
	Identity Manager and Identity Synchronization for Windows Functionality	143
	Password Changes on Active Directory	144
	Password Changes on Directory Server	144
	Password Changes and Provisions Originating From Identity Manager Administrator Interface	145
	Configuring Identity Manager and Identity Synchronization for Windows	145
	Setting Up Identity Manager 5.0 SP2	145
	Setting Up Identity Manager 5.0 SP1	147
	Configuring Identity Synchronization for Windows	148
C	Using Audit and Debug Logging to Isolate Problems	151
	Audit Logging and Action IDs	151
	Action Types	152
	Connector Layers: Accessor, Controller, and Agent	152
	Directory Server Plug-In	157
	Debug Logging	158
	Debug Logging in Java Components	158
	Debug Logging in the Installer	161
	Debug Logging in the Console	162

Windows NT Change Detection 162

Changing the Location of the Central Logs 165

Changing the Location of the Component Logs 165

Isolating Problems in Directory Server 166

Isolating Problems in Message Queue 167

Isolating Problems in Active Directory 167

Glossary 169

Index 179

Figures

FIGURE 2-1	Example Bank Architecture	24
FIGURE 2-2	shadowAccount Object Class	42
FIGURE 2-3	A List of SUL Parameters for Directory Server	48
FIGURE 3-1	Data Center Information for Directory Server	71
FIGURE 3-2	Data Center Information for Active Directory	72
FIGURE 3-3	Primary Installation of Identity Synchronization for Windows	75
FIGURE 3-4	Failover Installation While the Primary Installation Is Active	78
FIGURE 3-5	Primary Installation After Reinstalling the Identity Synchronization for Windows Plug-ins	79
FIGURE B-1	Password Synchronization and User Creation in an Identity Manager and Identity Synchronization for Windows Environment	143
FIGURE C-1	Action Paths	154

Tables

TABLE 2-1	Technical Requirements for Example Bank	25
TABLE 2-2	Example User Entries-SUL Matches	48
TABLE 2-3	Guidelines for Transitioning Contractor to Employee Accounts	66
TABLE 3-1	SSL Communication Between Components	94
TABLE C-1	Component-Level Debug Log Levels	160

Examples

EXAMPLE A-1	Example /etc/pam.conf File	138
-------------	----------------------------------	-----

Preface

The *Sun Java System Identity Synchronization for Windows Deployment Planning Guide* describes how to plan and implement a Sun Java System Identity Synchronization for Windows (Identity Synchronization for Windows) system.

Who Should Use This Book

This book is meant for system administrators who manage user identities on various directory resources. You must understand directory technologies and be familiar with directory servers, databases, and the Lightweight Directory Access Protocol (LDAP).

Once you understand the concepts described in this guide, you will be ready to plan and deploy an Identity Synchronization for Windows solution for your particular environment.

Before You Read This Book

Before trying to deploy the Identity Synchronization for Windows product, read the following publications:

- *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide*
- *Sun Java System Directory Server Enterprise Edition 6.0 Release Notes*

These publications explain Identity Synchronization for Windows product features, provide step-by-step instructions related to installing and configuring the product, and contain other important information that is not provided in this guide.

In addition, *read this guide* in its entirety before trying to deploy the Identity Synchronization for Windows product.

Books in This Documentation Set

The following table summarizes the books included in this documentation set.

TABLE P-1 Directory Server Enterprise Edition Documentation

Document Title	Contents
<i>Sun Java System Directory Server Enterprise Edition 6.0 Release Notes</i>	Contains the latest information about Directory Server Enterprise Edition, including known problems.
<i>Sun Java System Directory Server Enterprise Edition 6.0 Documentation Center</i>	Contains links to key areas of the documentation set.
<i>Sun Java System Directory Server Enterprise Edition 6.0 Evaluation Guide</i>	Introduces the key features of this release. Demonstrates how these features work and what they offer in the context of a fictional deployment that you can implement on a single system.
<i>Sun Java System Directory Server Enterprise Edition 6.0 Deployment Planning Guide</i>	Explains how to plan and design highly available, highly scalable directory services based on Directory Server Enterprise Edition. Presents the basic concepts and principles of deployment planning and design. Discusses the solution life cycle, and provides high-level examples and strategies to use when planning solutions based on Directory Server Enterprise Edition.
<i>Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide</i>	<p>Explains how to install the Directory Server Enterprise Edition software. Shows how to select which components to install, configure those components after installation, and verify that the configured components function properly.</p> <p>For instructions on installing Directory Editor, go to http://docs.sun.com/coll/DirEdit_05q1.</p> <p>Make sure you read the information in <i>Sun Java System Directory Server Enterprise Edition 6.0 Release Notes</i> concerning Directory Editor before you install Directory Editor.</p>
<i>Sun Java System Directory Server Enterprise Edition 6.0 Migration Guide</i>	Provides instructions for upgrading components from earlier versions of Directory Server, Directory Proxy Server, and Identity Synchronization for Windows.
<i>Sun Java System Directory Server Enterprise Edition 6.0 Administration Guide</i>	<p>Provides command-line instructions for administering Directory Server Enterprise Edition.</p> <p>For hints and instructions on using the Directory Service Control Center, DSCC, to administer Directory Server Enterprise Edition, see the online help provided in DSCC.</p> <p>For instructions on administering Directory Editor, go to http://docs.sun.com/coll/DirEdit_05q1.</p> <p>For instructions on installing and configuring Identity Synchronization for Windows, see Part II, “Installing Identity Synchronization for Windows,” in <i>Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide</i>.</p>

TABLE P-1 Directory Server Enterprise Edition Documentation (Continued)

Document Title	Contents
<i>Sun Java System Directory Server Enterprise Edition 6.0 Developer's Guide</i>	Shows how to develop server plug-ins with the APIs that are provided as part of Directory Server Enterprise Edition.
<i>Sun Java System Directory Server Enterprise Edition 6.0 Reference</i>	Introduces the technical and conceptual foundations of Directory Server Enterprise Edition. Describes its components, architecture, processes, and features. Also provides a reference to the developer APIs.
<i>Sun Java System Directory Server Enterprise Edition 6.0 Man Page Reference</i>	Describes the command-line tools, schema objects, and other public interfaces that are available through Directory Server Enterprise Edition. Individual sections of this document can be installed as online manual pages.
<i>Sun Java System Identity Synchronization for Windows 6.0 Deployment Planning Guide</i>	Provides general guidelines and best practices for planning and deploying Identity Synchronization for Windows

Related Documentation

Because you will be working with Sun Java System Directory Server and Sun Java System Message Queue, you might need to refer to their product documentation:

- Sun Java System Directory Server documentation
http://docs.sun.com/coll/DirectoryServer_04q2
- Sun Java System Message Queue documentation
http://docs.sun.com/app/docs/coll/MessageQueue_35_SP1

For information about the basic concepts of public-key cryptography; the Secure Sockets Layer (SSL) protocol; intranet, extranet, and Internet security; and the role of digital certificates in an enterprise, read the security-related appendixes in the *Managing Servers With iPlanet Console 5.0* manual.

For information about the Windows 2003 Server and Windows password policies, read the following Microsoft publications:

- *Using Secedit.exe to Force Group Policy to Be Applied Again - Windows 2000 Servers*
Microsoft KB #227448
- *A Description of the Group Policy Update Utility - Windows 2003 Servers* Microsoft KB #298444
- *Microsoft Knowledge Base Article 232690*

Typographic Conventions

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

TABLE P-2 Typographic Conventions

Typeface	Meaning	Example
AaBbCc123	The names of commands, files, and directories, and onscreen computer output	Edit your <code>.login</code> file. Use <code>ls -a</code> to list all files. <code>machine_name%</code> you have mail.
AaBbCc123	What you type, contrasted with onscreen computer output	<code>machine_name% su</code> Password:
<i>AaBbCc123</i>	A placeholder to be replaced with a real name or value	The command to remove a file is <code>rm filename</code> .
<i>AaBbCc123</i>	Book titles, new terms, and terms to be emphasized (note that some emphasized items appear bold online)	Read Chapter 6 in the <i>User's Guide</i> . <i>A cache</i> is a copy that is stored locally. Do <i>not</i> save the file.

Shell Prompts in Command Examples

The following table shows default system prompts and superuser prompts.

TABLE P-3 Shell Prompts

Shell	Prompt
C shell on UNIX and Linux systems	<code>machine_name%</code>
C shell superuser on UNIX and Linux systems	<code>machine_name#</code>
Bourne shell and Korn shell on UNIX and Linux systems	<code>\$</code>
Bourne shell and Korn shell superuser on UNIX and Linux systems	<code>#</code>
Microsoft Windows command line	<code>C:\</code>

Symbol Conventions

The following table explains symbols that might be used in this book.

TABLE P-4 Symbol Conventions

Symbol	Description	Example	Meaning
[]	Contains optional arguments and command options.	<code>ls [-l]</code>	The <code>-l</code> option is not required.
{ }	Contains a set of choices for a required command option.	<code>-d {y n}</code>	The <code>-d</code> option requires that you use either the <code>y</code> argument or the <code>n</code> argument.
<code>\${ }</code>	Indicates a variable reference.	<code>\${com.sun.javaRoot}</code>	References the value of the <code>com.sun.javaRoot</code> variable.
-	Joins simultaneous multiple keystrokes.	Control-A	Press the Control key while you press the A key.
+	Joins consecutive multiple keystrokes.	Ctrl+A+N	Press the Control key, release it, and then press the subsequent keys.
→	Indicates menu item selection in a graphical user interface.	File → New → Templates	From the File menu, choose New. From the New submenu, choose Templates.

Documentation, Support, and Training

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

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

Third-Party Web Site References

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

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

Searching Sun Product Documentation

Besides searching for Sun product documentation from the docs.sun.com web site, you can use a search engine of your choice by typing the following syntax in the search field:

search-term site:docs.sun.com

For example, to search for Directory Server, type the following:

"Directory Server" site:docs.sun.com

To include other Sun web sites in your search, such as java.sun.com, www.sun.com, and developers.sun.com, use sun.com in place of docs.sun.com in the search field.

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. To share your comments, go to <http://docs.sun.com> and click Send Comments. In the online form, provide the full document title and part number. The part number is a 7-digit or 9-digit number that can be found on the book's title page or in the document's URL. For example, the part number of this book is 819-4635.

Before You Begin Deployment

This chapter covers the following topics:

- [“Features of Identity Synchronization for Windows” on page 19](#)
- [“Pre-deployment Requirements and Guidelines” on page 20](#)

Sun Java™ System Identity Synchronization for Windows (Identity Synchronization for Windows) synchronizes user account information, including passwords, between Solaris™ Operating System (Solaris OS), by using Sun Java System Directory Server (Directory Server), and Windows (both Active Directory and Windows NT). Identity Synchronization for Windows helps build a scalable and security-enriched password synchronization solution for small, medium, and large enterprises.

This document provides guidance through case studies that lead to deploying the solution in your test and production environments.

Features of Identity Synchronization for Windows

The Identity Synchronization for Windows solution provides the following features:

- A robust and scalable solution that includes support for high availability
- Synchronization of account creation, modification, inactivation, and deletion between Active Directory and Directory Server, or Windows NT and Directory Server.
- Seamless integration with a disparate and proprietary directory source to synchronize native password changes

Pre-deployment Requirements and Guidelines

Before you begin deploying Identity Synchronization for Windows, you must be aware of the following deployment requirements and guidelines:

- **Synchronization direction of passwords.** If passwords are synchronized from Directory Server to Active Directory or in both directions, you must install High Encryption Pack on Windows 2000 to enable the 128-bit Secure Sockets Layer (SSL), which is required when setting passwords in Active Directory over Light Weight Directory Access Protocol (LDAP).
- **Synchronization of the creation of new users.** If Identity Synchronization for Windows does not synchronize the creation of new users, you must run the `idsync resync` command periodically to establish links between the newly created users. Changes to newly created users are not synchronized until the users are explicitly linked by running `idsync resync`.
- **Population size.** While Identity Synchronization for Windows places no upper limit on the number of users that can be synchronized, the total number of users impacts the deployment. The primary impact is on the `idsync resync` command, which must be run before you start synchronization. If more than 100,000 users are synchronized, run the `idsync resync` command in batches for optimal performance, and to limit the load on Sun Java System Message Queue.
- **Performance requirements.** The performance of Identity Synchronization for Windows is limited more by the synchronization rate than by the total number of users. The only exception to this requirement is when you run the `idsync resync` command.
- **Expected peak modification rate.** An Identity Synchronization for Windows deployment with Core and two connectors running on the same system can easily sustain a modification rate of 10 synchronizations per second. If the required synchronization rate exceeds this rate, higher performance can be achieved by distributing Identity Synchronization for Windows across multiple machines. For example, install the connectors on a separate machine from the Identity Synchronization for Windows Core.
- **Number of Windows domains to be synchronized.** If more than one Windows domain is to be synchronized, synchronize the `activedirectorydomainname`, the `USER_NT_DOMAIN_NAME` or both attributes to a Directory Server attribute to resolve ambiguity between Synchronization User List (SUL) definitions.
- **Number of preferred and secondary Directory Servers, hubs, and read-only master replicas in the deployment.** In a deployment with multiple Directory Servers, the Identity Synchronization for Windows Directory Server plug-in must be installed on each preferred and secondary Directory Server, each hub, and each read-only master replica. When configuring Identity Synchronization for Windows, one Directory Server is designated as the preferred Directory Server. The Directory Server Connector detects and applies changes to the preferred Directory Server while it is running. If this server is down, the Directory Server Connector can optionally apply changes to a secondary Directory Server. The Retro Changelog plug-in must be enabled on the preferred Directory Server, and this Directory Server should be on the same Local Area Network (LAN) as the Identity Synchronization for Windows Core.

- **Security.** If the Directory Server or Active Directory connectors connect to Directory Server or Active Directory over SSL, SSL must be enabled in these directories. If the connectors are configured to accept only trusted certificates, extra configuration steps must be followed to import the appropriate Certificate Authority certificates into the connectors' certificate databases. If SSL is required between the Directory Server plug-in and Active Directory, SSL must be enabled in Directory Server. The Certificate Authority certificate used to sign the Active Directory SSL certificate must also be imported in the Directory Server's certificate database.
- **Bidirectional account lockout and unlockout synchronization.** For Account Lockout and Unlockout to work correctly, set the symmetric password policy at both ends. For example, if the password policy for Active Directory signifies a permanent lockout then the same password policy should be set for Directory Server.
- **Bidirectional group synchronization.** When the Group Synchronization feature is enabled, the creation expression would be `uid=%uid%` or `cn=%cn%` in the Sun Java System Directory Server Criteria panel.

Case Study: Deploying in a Multimaster Replication Environment

The case study provided in this chapter explains how the company “Example Bank” implemented Identity Synchronization for Windows in a multimaster replication (MMR) environment. This case study includes information about the business imperatives, the technical requirements, and the implementation of Identity Synchronization for Windows.

This chapter covers the following topics:

- [“Example Bank Deployment Information” on page 23](#)
- [“Deploying the Solution” on page 28](#)

Example Bank Deployment Information

This section describes the architecture and technical requirements of Example Bank, the company in this case study. It also lists the Identity Synchronization for Windows features that are used in this case study.

Example Bank’s Existing Architecture

Example Bank’s infrastructure includes a Windows NT domain (EXBANK), an Active Directory domain (eb.com) with two domain controllers, and a two-way MMR Sun Java System Directory Server (dc=eb, dc=com) deployment. Example Bank has two main sites: one located in New York City and one in Los Angeles.

The following figure describes Example Bank’s deployment of its directory resources.

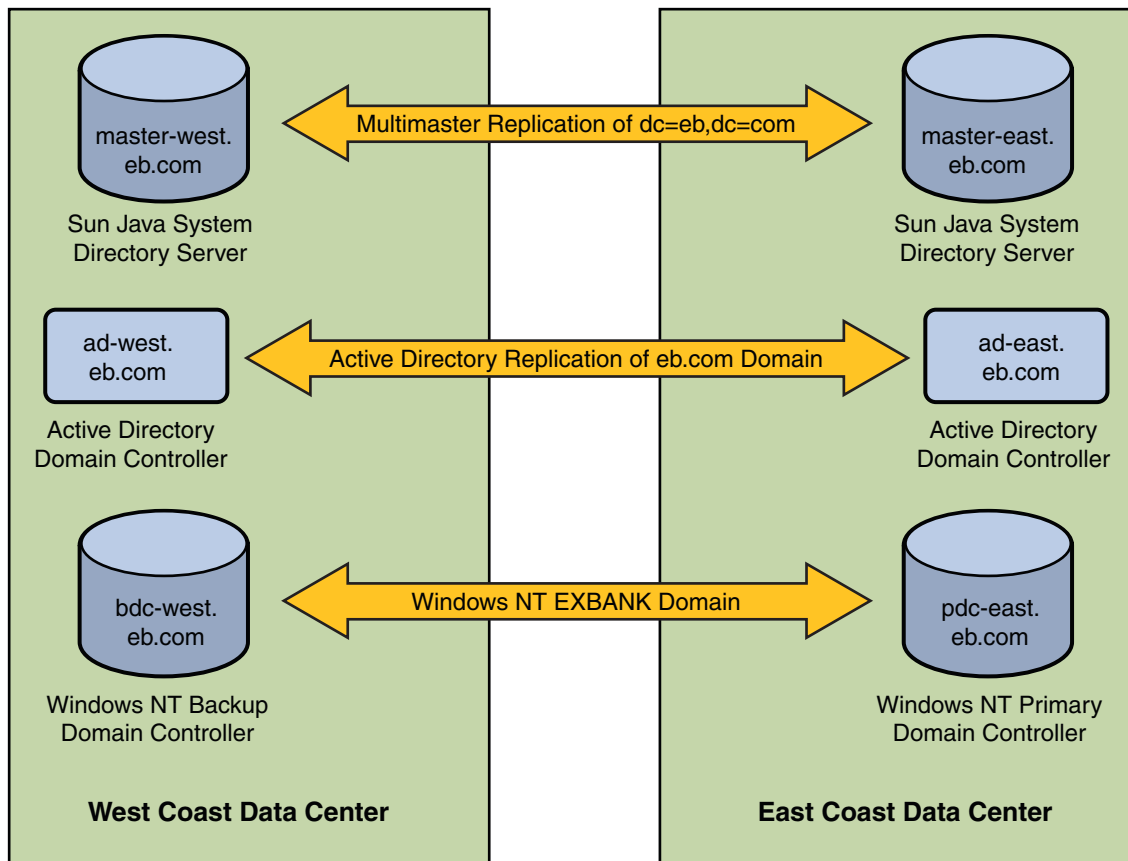


FIGURE 2-1 Example Bank Architecture

Directory Server Information

Sun Java System Directory Server is the corporate directory server that is used to control access to all web-based applications. Pluggable Authentication Module (PAM) for LDAP authenticates and manages passwords on the Solaris™ Operating System (Solaris OS) against Directory Server passwords. The two preferred Directory Servers manage a single root suffix, `dc=eb,dc=com`, and all users are stored in the `ou=people,dc=eb,dc=com` container with `uid` as the naming attribute. The directory servers, installed on Solaris systems, are running on separate machines: `master-east.eb.com` and `master-west.eb.com`.

Windows NT Information

The single Windows NT domain is called EXBANK. The Primary Domain Controller (PDC) runs on a `pdc-east.eb.com` machine in New York City. A backup domain controller (`bdc-west.eb.com`) runs on a machine located in Los Angeles. All Windows NT user accounts

have a Directory Server account with the exception of the built-in Windows NT accounts. The Windows NT `USER_NAME` attribute is the same as the Directory Server `uid` attribute.

Active Directory Information

The Active Directory deployment has a single domain, `eb.com`, with two domain controllers:

- `ad-east.eb.com` (in New York City)
- `ad-west.eb.com` (in Los Angeles)

In this deployment, `ad-west.eb.com` is the PDC Flexible Single-Master Operation (FSMO) role owner.

Users are stored in two separate organizations corresponding to the two sites:

- `ou=east,dc=example,dc=com`
- `ou=west,dc=example,dc=com`

Example Bank is in the process of migrating users from Windows NT to Active Directory. Each employee has a Windows NT or Active Directory account. The migration of the users is based (in phases) on the employees' last names. Every week Example Bank moves users whose last name begins with the next letter of the alphabet. Currently, the company has migrated employees whose last names begin with letters A through F.

For users who have Directory Server accounts, the Active Directory `samaccountname` attribute stores the `uid`. When a user account is migrated from Windows NT, the user keeps the same login. That is, the Active Directory `samaccountname` attribute of the new user is the same as the Windows NT `USER_NAME` attribute.

Example Bank's Technical Requirements

The following table describes Example Bank's technical requirements.

TABLE 2-1 Technical Requirements for Example Bank

Requirement	Comments
Users' Windows passwords should be synchronized with their Directory Server passwords.	Identity Synchronization for Windows can synchronize a user's existing Directory Server password with the user's Active Directory password.

TABLE 2-1 Technical Requirements for Example Bank (Continued)

Requirement	Comments
<p>Users must be able to continue to change passwords using native mechanisms in Windows NT or Active Directory.</p> <p>That is, use the CTRL+ALT+DEL key sequence on Windows systems and the passwd command on Solaris systems.</p>	<p>Identity Synchronization for Windows supports capturing native password changes in Directory Server, Active Directory, and Windows NT. Users can continue to change their passwords. They are not required to change passwords at a central location (for example, a web form).</p> <p>For native Solaris OS password changes to be propagated to Active Directory, Solaris must be configured to use PAM for LDAP. This feature is discussed in detail in Appendix A.</p> <p>Note: You can set passwords in Directory Server by passing in a pre-hashed password value. However, Identity Synchronization for Windows cannot synchronize passwords from Directory Server to Windows if the password is pre-hashed. Thus, do not use a pre-hashed password value because it circumvents password policy and password history.</p>
<p>Contractors might have accounts on Active Directory, Windows NT, or Directory Server but their accounts should <i>not</i> be synchronized.</p> <p>Contractors can be identified using their login name, which starts with c-. The uid, samaccountname, and USER_NAME attributes start with c- for contractors only.</p>	<p>The Synchronization User List (SUL) feature of Identity Synchronization for Windows can be used to eliminate user accounts that must not be synchronized. Part of an SUL's configuration includes an LDAP-like filter.</p> <p>A contractor's login name begins with a c-.</p> <p>To exclude contractors use filters:</p> <ul style="list-style-type: none"> ■ Active Directory SUL filter is (!(samaccountname=c-*)) ■ Windows NT SUL filter is (!(USER_NAME=c-*)) ■ Directory Server SULs include (!(uid=c-*)) as a part of their filters. Directory Server SUL filters specify whether the user is using Windows NT or Active Directory.
<p>Changes to login IDs (USER_NAME, samaccountname, and uid) and full names (USER_FULL_NAME, cn, and cn) should be synchronized when they change.</p>	<p>Identity Synchronization for Windows synchronizes other attributes between Windows and Directory Server. It uses a unique Globally Unique Identifier (GUID) stored in the Directory Server entry to correlate users. It can synchronize attributes, such as uid and cn, that are otherwise used as keys.</p> <p>Note: The SUL filter applies to all operations, and no other changes for contractors are synchronized.</p>
<p>When a new user is created in Active Directory (except for contractors), a corresponding user should be created in Directory Server.</p>	<p>Using Identity Synchronization for Windows, you can choose how creations of new users will be synchronized. Identity Synchronization for Windows can synchronize new user entries from Windows to Directory Server, from Directory Server to Windows, both ways, or not at all. The same SUL filter also excludes contractors from being created in Directory Server.</p>

TABLE 2-1 Technical Requirements for Example Bank (Continued)

Requirement	Comments
New users created in Directory Server must not have an Active Directory account created automatically.	Identity Synchronization for Windows can synchronize new user accounts from Windows to Directory Server, Directory Server to Windows, or not at all.
Entries deleted in Directory Server or Active Directory must not be deleted in the corresponding data source.	<p>Identity Synchronization for Windows can synchronize account deletions from Active Directory to Directory Server, Directory Server to Active Directory, both ways, or not at all. Identity Synchronization for Windows does not synchronize bidirectional account deletions, that is, from Active Directory to Directory Server <i>and</i> from Directory Server to Active Directory.</p> <p>Note: Identity Synchronization for Windows does not synchronize account deletions with Windows NT.</p>
Accounts that are made inactive (disabled) in Directory Server or Active Directory should be deactivated in the corresponding data source.	<p>Identity Synchronization for Windows can synchronize deactivated accounts from Active Directory to Directory Server, Directory Server to Active Directory, both ways, or not at all.</p> <p>You can configure how Identity Synchronization for Windows detects account inactivations in Directory Server. Example Bank is not using a third-party application such as Sun Java System Access Manager to control access to its directory server, so the default account deactivation setting for interoperating with the Directory Server tools suffices.</p> <p>Note: Identity Synchronization for Windows does not synchronize account inactivations with Windows NT.</p>
After a user has been migrated from Windows NT to Active Directory, changes should be synchronized with the Active Directory account but not the Windows NT account.	Identity Synchronization for Windows stores a GUID in each Directory Server entry that it synchronizes (in the <code>dspswuserlink</code> attribute), to synchronize changes made in Directory Server to Active Directory and Windows NT. If an account is moved from Windows NT to Active Directory, the <code>dspswuserlink</code> attribute in the corresponding Directory Server entry should be updated. To update the entry in Directory Server, remove the attribute and then run the <code>idsync resync -f <linking-file></code> command again.
Users should continue to synchronize even when one preferred Directory Server fails.	The failover feature in Identity Synchronization for Windows happens seamlessly when the preferred Directory Server fails during synchronization between Active Directory and Directory Server.
Account lockout synchronization must happen between Directory Server and Active Directory sources.	In Identity Synchronization for Windows, the accounts locked in Directory Server are also locked automatically in Active Directory. The reverse is also true.

TABLE 2-1 Technical Requirements for Example Bank (Continued)

Requirement	Comments
The users under various groups, such as employee, supervisor, manager, and contractor, should be synchronized across Directory Server and Active Directory with their group memberships intact.	The static group synchronization in Identity Synchronization for Windows synchronizes the users across the directory servers with their group memberships intact.

Identity Synchronization for Windows Features in This Case Study

The following Identity Synchronization for Windows features are used in this case study:

- Synchronizing with multiple Windows domains
- Synchronizing in a two-way MMR environment
- Synchronizing multiple object classes
- Deploying over a wide area network (WAN)
- Integration with PAM LDAP
- Migrating users between Windows domains, (from Windows NT to Active Directory)
- Excluding users from synchronization using an SUL filter
- Synchronizing a hierarchical directory information tree (DIT) with a flat DIT
- Creating attribute default values
- Failover support across multiple preferred Directory Servers in a replicated environment
- Account lockout synchronization
- Static group synchronization

Deploying the Solution

This section provides instructions for the following tasks:

- [“Creating a Special Active Directory User for Identity Synchronization for Windows” on page 29](#)
- [“Configuring the Identity Synchronization for Windows Core” on page 30](#)
- [“Migrating Users From Windows NT to Active Directory” on page 61](#)

Creating a Special Active Directory User for Identity Synchronization for Windows

Example Bank creates a special user that Identity Synchronization for Windows uses when connecting to Active Directory. This user is created in the `cn=Users` container in the `eb.com` domain. After the user is created, a minimum set of administration rights is assigned to this user.

Note – Identity Synchronization for Windows automatically creates a similar user with limited privileges in Directory Server. This user is created as the `uid=PSWConnector,<synchronized suffix\>user`.

▼ To Assign Administration Rights to the Special User

- 1 In the Tree pane of the Active Directory Users and Computers window, right-click the `eb.com` container icon.
- 2 From the All Tasks menu, choose **Delegate Control**.
The Delegation of Control Wizard is displayed.
- 3 In the Selected User and Groups list, select the special user and click **Next**.
- 4 In the Tasks to Delegate window, select **Create a Custom Tasks to Delegate**, and click **Next**.
- 5 In the Only the Following Objects in the Folder section, select **User Objects**.
Identity Synchronization for Windows manages only User objects, so it is sufficient to delegate control of these objects.
- 6 In the Show These Permissions list of the Permissions window, select these options:
 - General
 - Property-Specific
 - Full ControlBecause Example Bank requires the synchronization of users from Directory Server to Active Directory, the special user is given full control of user objects in the `eb.com` domain.

Note – If you specify a user with default Active Directory permissions, some operations will succeed, for example, an `idsync resync` operation from Active Directory to Directory Server. Other operations, such as detecting and applying changes in Active Directory, can fail abruptly.

If Example Bank is synchronizing the deletions from Active Directory to Directory Server, even Full Control is insufficient. You must use a Domain Administrator account to detect account deletions in Active Directory.

Configuring the Identity Synchronization for Windows Core

The Identity Synchronization for Windows Core components are installed on `master-east.eb.com`, the machine on which the preferred Directory Server is running. After the Core components are installed, complete the following configuration procedures:

- [“Configuring Directory Sources” on page 30](#)
- [“Configuring the Synchronization Settings” on page 36](#)
- [“SUL_NT” on page 43](#)
- [“Installing the Connectors and Directory Server Plug-Ins” on page 51](#)
- [“Running `idsync resync`” on page 52](#)
- [“Configuration and Installation Summary” on page 59](#)

Configuring Directory Sources

This section explains how to configure the following directory sources:

- [“Configuring the Sun Java System Directory Server Source” on page 30](#)
- [“Configuring the Active Directory Source” on page 32](#)
- [“Configuring the Windows NT Source” on page 35](#)

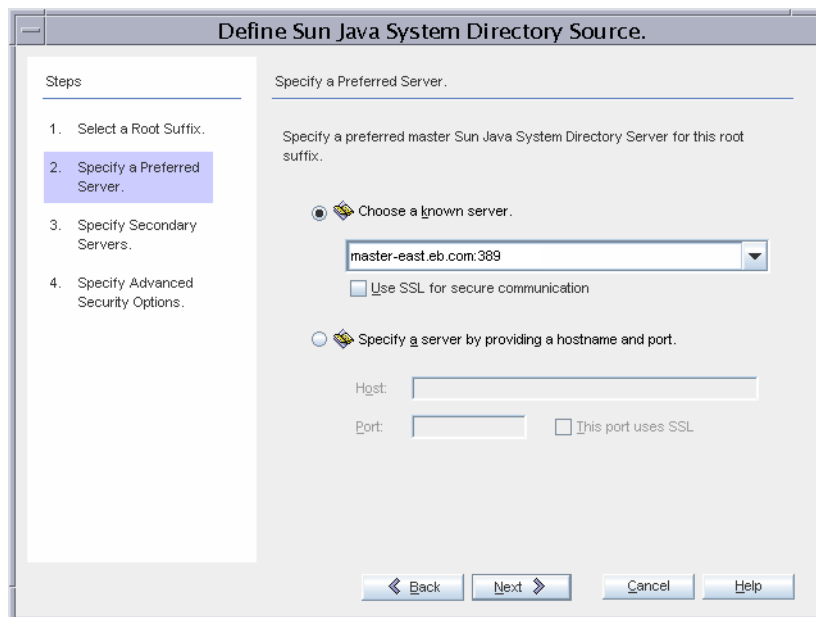
Configuring the Sun Java System Directory Server Source

When configuring the Directory Server source, the preferred Directory Server is set to `master-east.eb.com`. The Directory Server Connector uses this Directory Server to detect and update changes that require synchronization with Active Directory and Windows NT. Alternatively, the `master-west.eb.com` domain could have been selected. However, Directory Server Connector performance is better when connecting to a local Directory Server instead of a Directory Server located over a wide area network (WAN).

Note – When the password modification settings are changed, the Console automatically enables the SSL option, which is required while synchronizing from Directory Server to Active Directory.

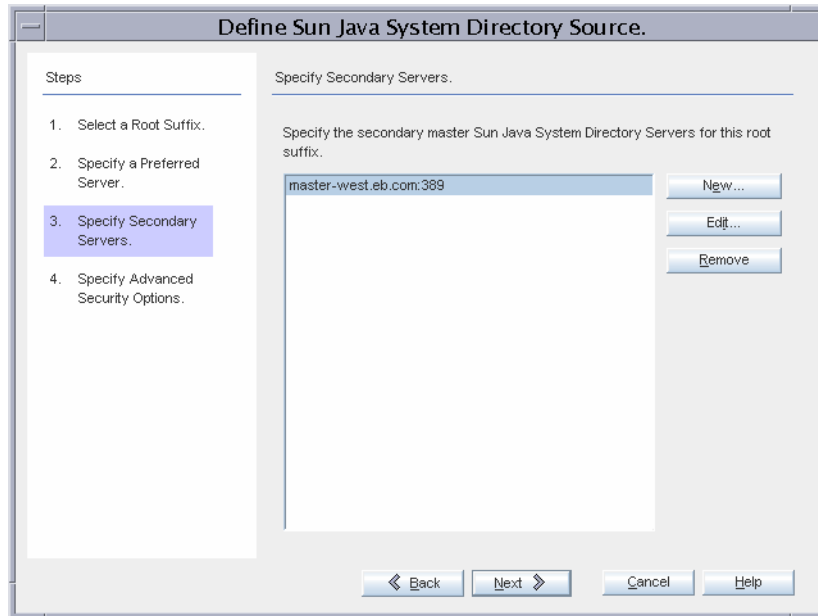
▼ To Specify the Preferred and Secondary Directory Servers

- 1 In the Directory Sources window of the Identity Synchronization for Windows Console (Console), click New Sun Directory.
- 2 In the Define Sun Java System Directory Source dialog box, select Specify a Preferred Server.



- 3 Select Choose a Known Server and then choose a preferred Directory Server from the drop-down menu, in this case, `master-east.eb.com`.
- 4 (Optional) Select Specify Secondary Servers to select a secondary Directory Server, in this case, `master-west.eb.com`.

If `master-east.eb.com` is unavailable, the Directory Server Connector synchronizes changes made at Active Directory to `master-west.eb.com`.



Configuring the Active Directory Source

The Active Directory global catalog information enables the Identity Synchronization for Windows Console to learn the Active Directory configuration. In this case study, the global catalog is running on `ad-west.eb.com`. By default, the Console auto-populates the User DN field with the Administrator DN, `cn=Administrator,cn=user,dc=eb,dc=com`. However, you need to change this field to the special Identity Synchronization for Windows user that was created earlier, `cn=iswUser,cn=Users,dc=eb,dc=com`.

▼ To Specify Information in the Global Catalog and for the Active Directory Domain

- 1 In the Console, in the Directory Sources window, click **New Active Directory Source**.
The Windows Global Catalog dialog box is displayed.
- 2 Type the fully qualified name in the **Host** field, in this example, `ad-west.ed.com`.
- 3 Change the default User DN (`cn=Administrator`) to the DN `cn=iswUser,cn=Users,dc=eb,dc=com`.
- 4 Type the password and click **OK**.

Windows Global Catalog

The system requires a global catalog from which to discover schema and topology information about the Windows domain. Specify the host and access credentials for the global catalog in your Windows domain.

Host:
 Enter a fully qualified name, for example, host.example.com.

☐ This port uses SSL

Directory Source Credentials:

User DN:
 Password:

OK Cancel Help

5 Provide credentials for the Active Directory domain, then click Next.

The Active Directory Connector uses the same Identity Synchronization for Windows special user credentials to connect to Active Directory that you provided when connecting to the global catalog.

Define an Active Directory Source.

Steps

1. Select a Domain.
2. Specify Credentials.
3. Specify a Domain Controller.
4. Specify Failover Controllers.
5. Specify Advanced Security Options.

Specify Credentials.

Specify credentials for accessing user entries in all servers in this directory source.

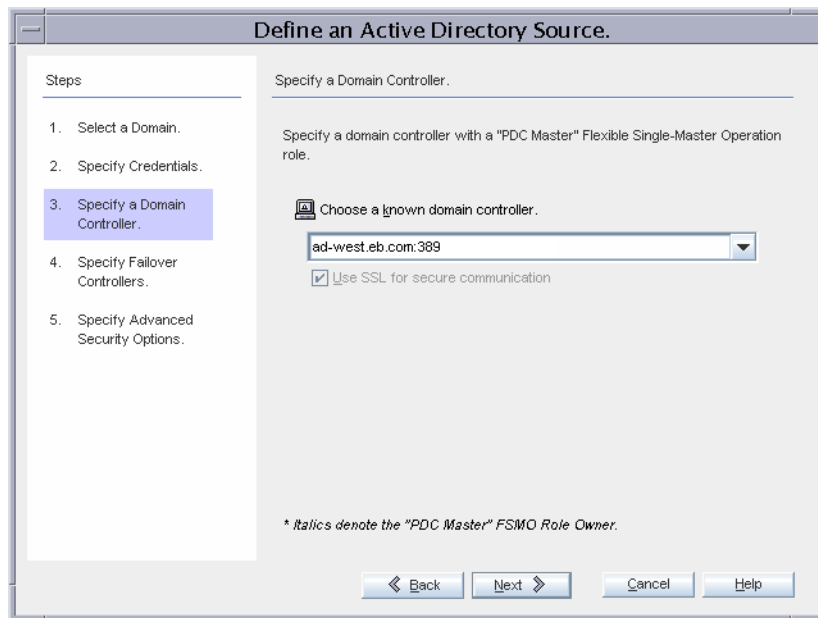
User DN:
 Password:

Back Next Cancel Help

6 Specify the PDC FSMO role owner domain controller.

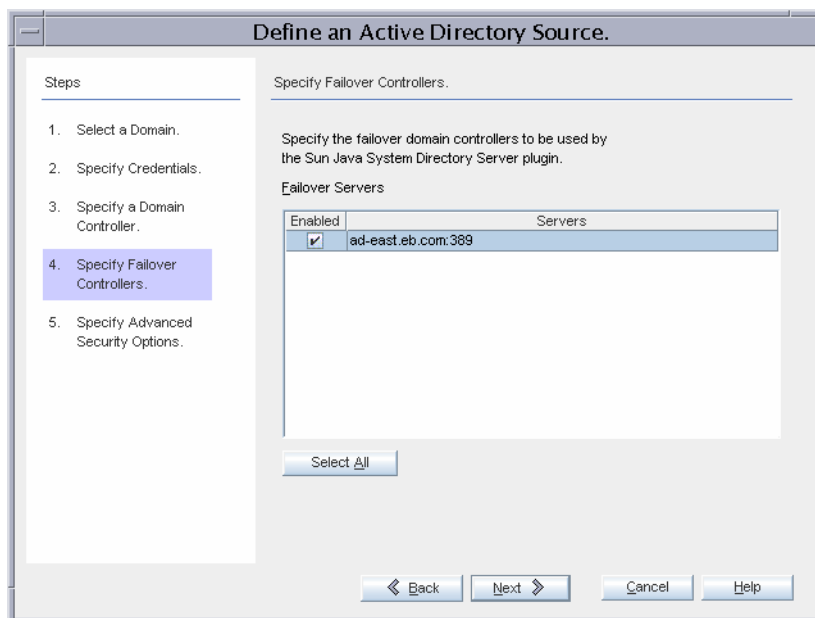
The `ad-west.eb.com` domain controller is the PDC FSMO role owner. Certain changes (for example, password modifications) made at other domain controllers are replicated immediately to this domain controller. The Active Directory Connector communicates with this domain controller so that changes made at any Active Directory domain controller can be synchronized immediately to Directory Server. This Active Directory replication can take several minutes.

The Active Directory Connector for this domain is installed on the same machine where Identity Synchronization for Windows Core is installed, on `master-east.eb.com`. The connector communicates over the WAN with `ad-west.eb.com`. Active Directory Connector performs better across WAN than the Directory Server Connector because Active Directory Connector performs fewer directory searches to detect changes.



7 Specify one or more failover domain controllers for on-demand password synchronization, in this case, `ad-east.eb.com`.

If `ad-west.eb.com` is unavailable, the Directory Server plug-in performs on-demand password synchronization against `ad-east.eb.com`.

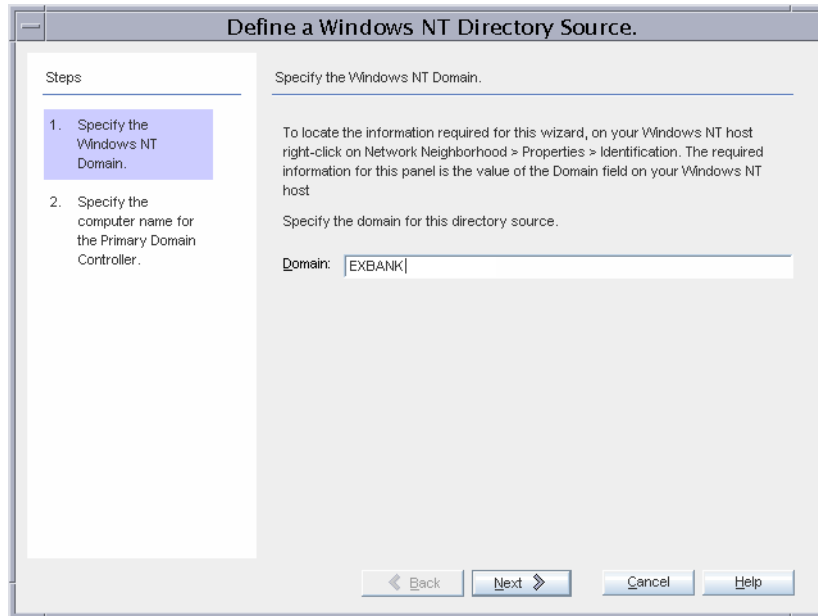


Configuring the Windows NT Source

After the Directory Server and the Active Directory sources are configured, configure the Windows NT domain.

▼ To Specify the Windows NT Domain

- 1 In the Console, in the Directory Sources window, click **New Windows NT Directory Source**. The Define a Windows NT Directory Source dialog box is displayed.



- 2 **Select Specify the Windows NT Domain, type the Windows NT domain, in this case, EXBANK, and click Next.**
- 3 **Type the Primary Domain Controller of the EXBANK domain.**
The NETBOIS name of the Primary Domain Controller is pdc-east. The fully qualified name of this host is pdc-east.eb.com.

Configuring the Synchronization Settings

After each directory source is configured, the synchronization parameters are configured to match Example Bank's requirements as explained in these section:

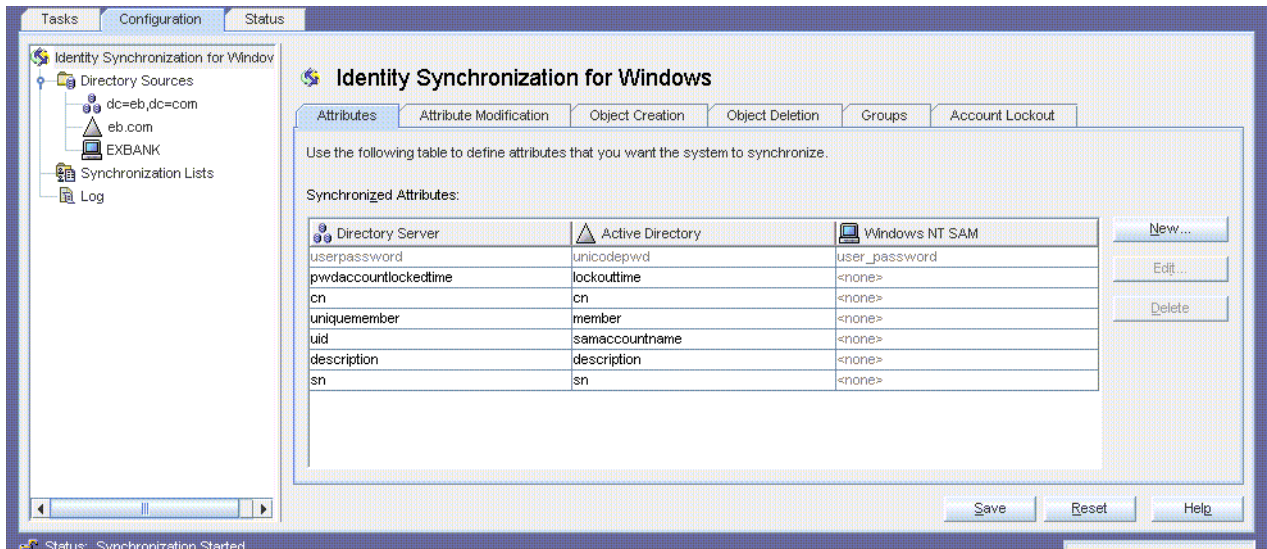
- [“Configuring the Attributes Settings” on page 36](#)
- [“Configuring the Attribute Modification Settings” on page 37](#)
- [“Configuring the Object Creation Settings” on page 38](#)
- [“Configuring the Group Synchronization Settings” on page 39](#)
- [“Configuring the Account Lockout Synchronization Settings” on page 40](#)

Configuring the Attributes Settings

The Attributes settings reflect Example Bank's requirement to synchronize changes to a user's password, full name, and login. The `destinationindicator <-\> activedirectorydomainname <-\> user_nt_domain_name` mapping displays because it synchronizes multiple Windows domains.

▼ To Configure the Attribute Settings

- 1 In the Console, click the Configuration tab, then click the Attributes tab.
- 2 Under Synchronized Attributes, enter the attributes that Example Bank requires to synchronize with Directory Server.



Note – Mapping an attribute to the synthetic `activedirectorydomainname` or `user_nt_domain_name` attribute is not unique to deployments that have both Active Directory and Windows NT domains. The same approach is taken in homogeneous Windows environments that have multiple Active Directory or Windows NT domains, where the `destinationindicator` attribute is mapped to `activedirectorydomainname` or `user_nt_domain_name`.

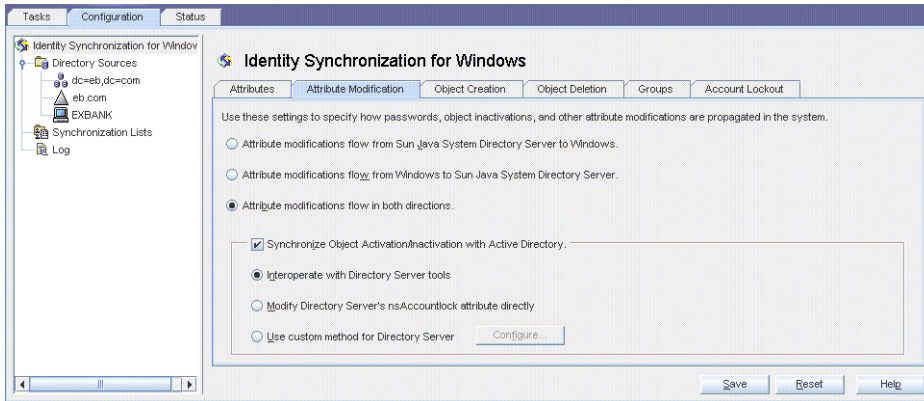
Configuring the Attribute Modification Settings

The Attribute Modification settings reflect Example Bank's requirements to synchronize the attribute changes and account deactivations, bidirectionally, between the Active Directory and Directory Server sources.

▼ To Configure the Attribute Modification settings

- 1 In the Identity Synchronization for Windows Console, click the Configuration tab, then click the Attribute Modification tab.

- 2 **Select Attribute Modifications Flow in Both Directions.**
- 3 **Select the Synchronize Object Activation/Inactivation with Active Directory check box and select Interoperate With Directory Server Tools.**



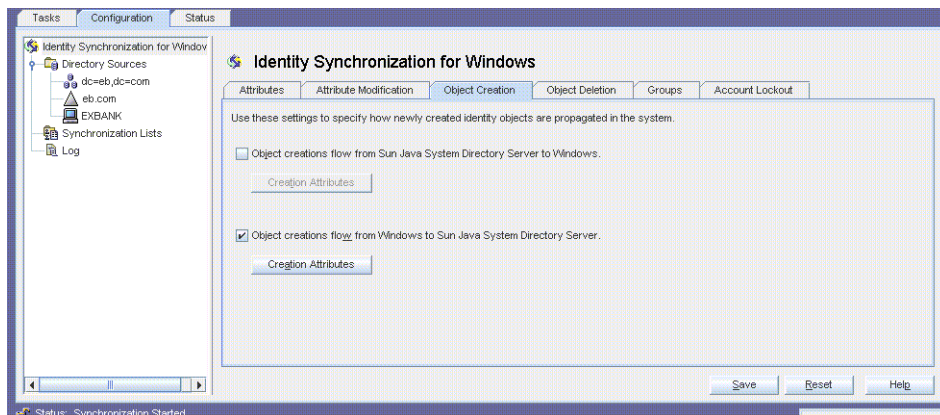
Configuring the Object Creation Settings

The Object Creation settings reflect Example Bank's requirement to only synchronize user creations from Active Directory to Directory Server.

The Object Creation settings apply to both Active Directory and Windows NT because Example Bank has an environment with both the systems. New users in Active Directory and Windows NT are synchronized with Directory Server. Example Bank is migrating all Windows NT users to Active Directory; so no new users will be created in Windows NT.

▼ To Configure the Object Creation Settings

- 1 **In the Console, click the Configuration tab, then click the Object Creation tab.**
- 2 **Select the Object Creations Flow From Windows to Sun Java System Directory Server check box.**



Note – To synchronize object deletions, click the Object Deletion tab and select Object Deletions Flow From Windows to Sun Java System Directory Server check box.

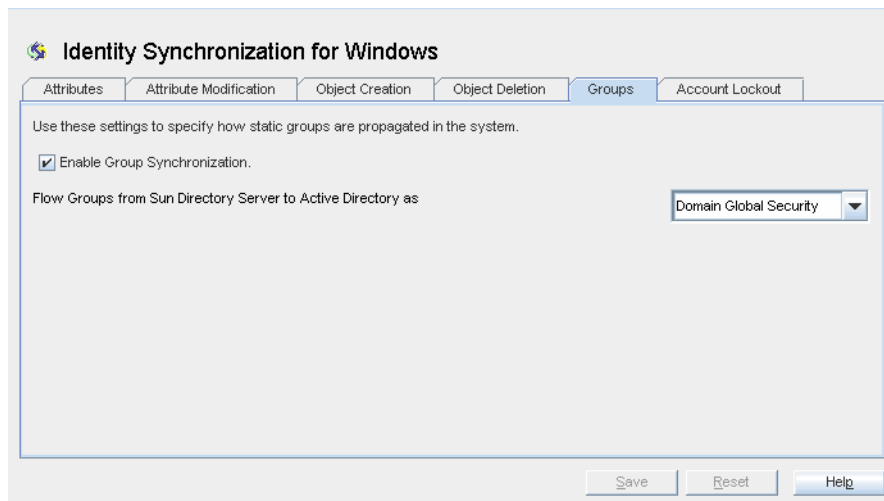
Configuring the Group Synchronization Settings

You can create or delete a group, and associate or disassociate users with that group in a directory environment. If Group Synchronization is enabled, the changes that you make in one directory environment automatically propagate to the other directory environment. All the users are synchronized across the directory servers with their group membership intact.

Note – When Group Synchronization is enabled, the `uniquemember` Directory Server attribute and the `member` attribute Active Directory attribute are internally mapped.

▼ To Configure the Group synchronization Settings

- 1 In the Console, click the Configuration tab, then click the Groups tab.
- 2 Select the Enable Group Synchronization check box.



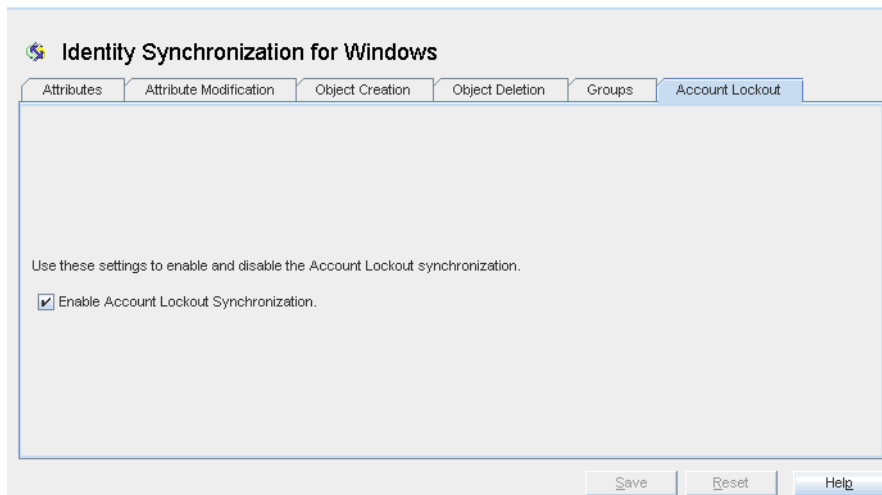
- 3 From the drop-down menu, choose Domain Global Security or Domain Global Distribution to propagate groups from Sun Directory Server to Active Directory.

Configuring the Account Lockout Synchronization Settings

In Identity Synchronization for Windows, account lockout and unlockout are synchronized between the Directory Server and Active Directory sources.

▼ To Configure the Account Lockout Synchronization Settings

- 1 In the Console, click the Configuration tab, then click the Account Lockout tab.
- 2 Select the Enable Account Lockout Synchronization check box.



Adding the shadowAccount Object Class

When configuring Identity Synchronization for Windows to interoperate with PAM LDAP on Solaris systems, select and then add the `shadowAccount` object class as an auxiliary object class for synchronization. When a new user is created in Active Directory, and that user is synchronized to Directory Server, the user entry includes the `shadowAccount` object class, which is required by PAM LDAP.

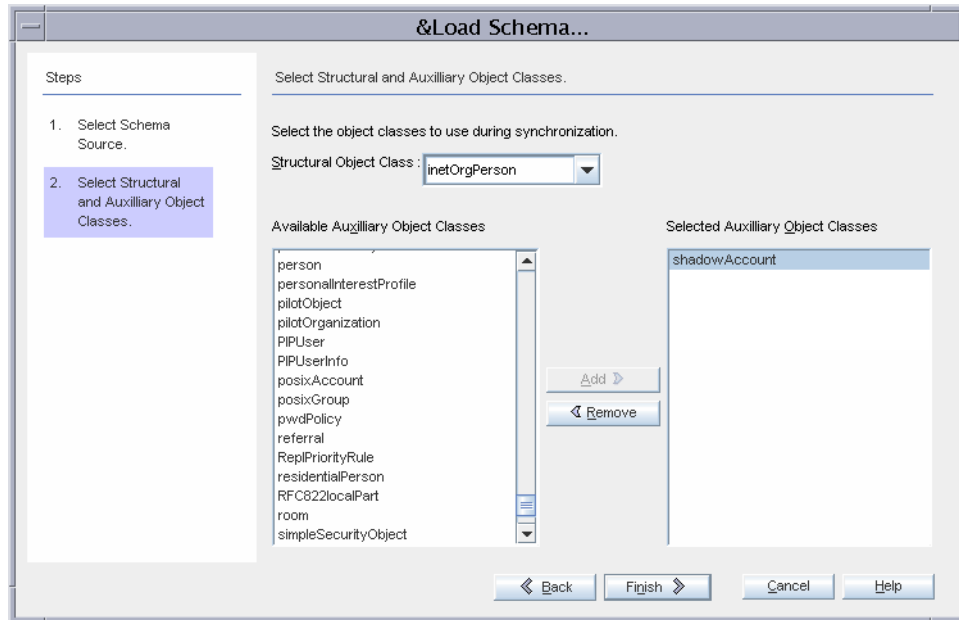


FIGURE 2-2 shadowAccount Object Class

Configuring the Creation Attributes

Use the Creation Attribute Mappings and Values dialog box to configure additional attributes to be synchronized when an entry is created.

▼ To Configure the Creation Attributes

- 1 Click **Creation Attributes** under the **Object Creation** tab.
- 2 Provide a mapping or default value for **sn**, a mandatory attribute for the **inetOrgPerson** object class.

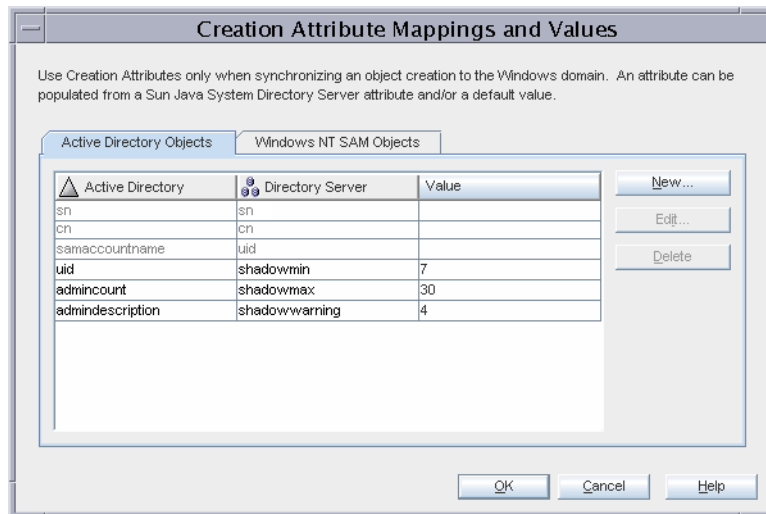
Active Directory has a corresponding attribute **sn**. However, Windows NT does not have an equivalent attribute, so the special **** NO VALUE **** value is provided. Because Example Bank's requirements do not include creating users in Windows NT, this value does not appear in any of the user entries. This value is only provided to conform to the Console's validations.

Configure the **shadowmin**, **shadowmax**, and **shadowwarning** attributes, which are used for PAM LDAP.

- A **shadowmin** value of 7 implies that a user must wait seven days from the time the password has changed before changing it again.
- A **shadowmax** value of 30 implies that the user must change the password at least every 30 days.

- A shadowwarning value of 4 implies that the user is warned that the password must be changed four days before the password expires.

Directory Server attributes that are grayed-out are mandatory creation attributes. The `inetOrgPerson` object class has `cn` and `sn` as mandatory attributes, and the `shadowAccount` object class has `uid` as a mandatory attribute.



Configuring the Synchronization User Lists

Example Bank requires at least two Synchronization User Lists (SULs) for each of the Windows domains (Windows NT and Active Directory). However, Example Bank wants to synchronize two disjoint organizational units in Active Directory (`ou=east` and `ou=west`), so two SULs are configured for the Active Directory domain.

SUL_NT

This SUL provides details required to synchronize users between Windows NT and Directory Server.

When detecting changes in Windows NT, contractors are excluded from synchronization by the `(!(USER_NAME=c-*))` filter because each contractor `USER_NAME` value starts with `c-`.

Define a Synchronization User List.

Steps

1. Specify a Name.
2. Specify the Windows Criteria.
3. Specify the Sun Java System Directory Server Criteria.

Specify the Windows Criteria.

Directory Source:

User Set Domain

Base DN:

Filter:

Creation Expression:
For example, cn=%cn%

The base DN of all synchronized users is, `ou=people,dc=eb,dc=com`. The Creation Expression, `uid=%uid%,ou=people,dc=eb,dc=com`, implies that new users will be created in the same container with `uid` as the naming attribute.

The filter for this SUL includes two components:

- `(!(uid=c-*))` excludes contractors from synchronization.
- `(destinationindicator=EXBANK)` allows the Directory Server Connector to determine the Windows domain where the user resides. (when it detects a change to the user)

Define a Synchronization User List.

Specify the Sun Java System Directory Server Criteria.

Steps

1. Specify a Name.
2. Specify the Windows Criteria.
3. Specify the Sun Java System Directory Server Criteria.

Directory Source: dc=eb,dc=com

User Set Domain

Base DN: ou=people,dc=eb,dc=com Browse...

Filter: (&((uid=c-*)(destinationindicator=EXBANK))

Creation Expression: uid=%uid% ,ou=isw_data
For example, cn=%cn%

Back Finish Cancel Help

SUL_AD_EAST

This SUL provides details required to synchronize users between the Active Directory ou=east , dc=eb , dc=com container and Directory Server.

The SUL settings imply that all users under ou=east , dc=eb , dc=com will be synchronized from Active Directory to Directory Server except users whose samaccountname starts with c - (contractors).

Define a Synchronization User List.

Specify the Windows Criteria.

Steps

1. Specify a Name.
2. Specify the Windows Criteria.
3. Specify the Sun Java System Directory Server Criteria.

Directory Source: eb.com

User Set Domain

Base DN: OU=east,DC=eb,DC=com Browse...

Filter: (!(samaccountname=c-*))

Creation Expression: ,DC=iswqa095,DC=com
For example, cn=%cn%

Back Next Cancel Help

For the SUL_AD_EAST list, only users under the `ou=east,dc=eb,dc=com` organizational unit will be synchronized. The filter for this SUL includes two components:

- `(!(uid=c-*))` excludes contractors from synchronization
- `destinationindicator=eb.com` allows the Connector to determine the Windows domain where the user resides.

Define a Synchronization User List.

Steps

1. Specify a Name.
2. Specify the Windows Criteria.
3. Specify the Sun Java System Directory Server Criteria.

Specify the Sun Java System Directory Server Criteria.

Directory Source: `dc=eb,dc=com`

User Set Domain

Base DN: `ou=people,dc=eb,dc=com` Browse...

Filter: `(&(!(uid=c-*))(destinationindicator=india.sun.com))`

Creation Expression: `uid=%uid%,ou=isw_data`
For example, `cn=%cn%`

Back Finish Cancel Help

SUL_AD_WEST

This SUL provides details required to synchronize users between the Active Directory `ou=west,dc=eb,dc=com` container and Directory Server.

The SUL settings imply that all users under `ou=west,dc=eb,dc=com` will be synchronized from Active Directory to Directory Server except users whose `samaccountname` starts with `c-` (contractors).

Define a Synchronization User List.

Steps

1. Specify a Name.
2. Specify the Windows Criteria.
3. Specify the Sun Java System Directory Server Criteria.

Specify the Windows Criteria.

Directory Source:

User Set Domain

Base DN:

Filter:

Creation Expression: ,dc=iswqa095,dc=com
For example, cn=%cn%

Resolving Issues With Multiple SULs

The SUL_AD_WEST settings are identical to the settings in SUL_AD_EAST.

Define a Synchronization User List.

Steps

1. Specify a Name.
2. Specify the Windows Criteria.
3. Specify the Sun Java System Directory Server Criteria.

Specify the Sun Java System Directory Server Criteria.

Directory Source:

User Set Domain

Base DN:

Filter:

Creation Expression: ,ou=isw_data
For example, cn=%cn%

When the configuration is saved, this error message is displayed:

“Synchronization User Lists: SUL_AD_EAST and SUL_AD_WEST contain duplicate directory source, dc=eb, dc=com, location, ou=people, dc=eb, dc=com, and filter, (&(! (uid=c-*) (destinationindicator=eb.com)).”

When there are multiple SULs, Identity Synchronization for Windows requires that the base DN or the filter is unique. When Identity Synchronization for Windows detects a change to a user, it iterates through the list of SULs until it matches a user. The user entry is compared first with SUL_NT. If it fails to match, it is compared with SUL_AD_WEST. If it fails to match again, it is compared with SUL_AD_EAST.

For summary information about the three defined SULs, click Resolve Domain Overlap.

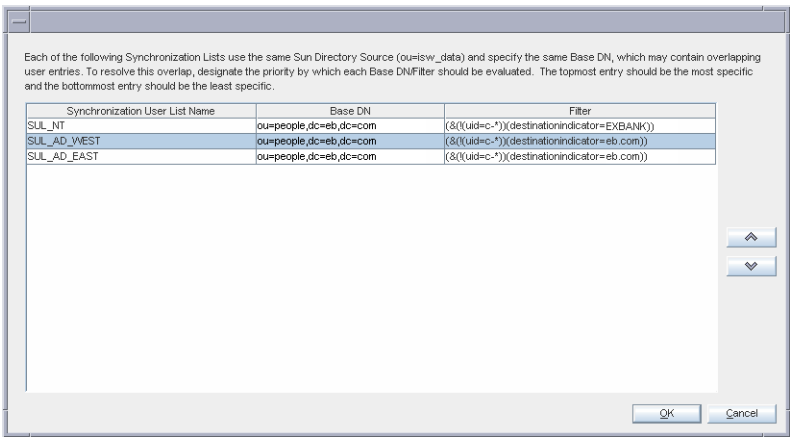


FIGURE 2-3 A List of SUL Parameters for Directory Server

The following table shows examples of user entries and SUL matches.

TABLE 2-2 Example User Entries-SUL Matches

User Entry	Matched SUL
dn: uid=someuser,ou=people,dc=eb,dc=com destinationindicator: EXBANK	SUL_NT
dn: uid=someuser,ou=people,dc=eb,dc=com destinationindicator: eb.com	SUL_AD_WEST
dn: uid=c-somecontractor,ou=people,dc=eb,dc=com destinationindicator: eb.com	<none>Because uid=c-*

TABLE 2-2 Example User Entries-SUL Matches (Continued)

User Entry	Matched SUL
dn: uid=someuser,ou=people,dc=eb,dc=com destinationindicator:	<none>Because no destinationindicator attribute
dn: uid=someuser,ou=otherpeople,dc=eb,dc=com	<none>Because it does not match any base DN's

The program will not find a match for the SUL_AD_EAST entry because SULs are evaluated from top to bottom. The program will encounter and evaluate the base DN and filter for the SUL_AD_WEST entry first, which is located immediately before the SUL_AD_EAST entry. Consequently, Console requires SULs to have different base DN's or filters.

A SUL is used primarily when a change to a user is detected. Evaluating the SUL in which the user resides determines the remote Windows domain or Directory Server the user should be synchronized with. When a change is detected to a user, it must be sent to the appropriate place. The information in an SUL is used in two instances at the destination resource where the change is applied:

- When the `idsync resync -f` command is executed to establish links between existing users, by default Identity Synchronization for Windows will only link to a Directory Server user that is in the same SUL as the Active Directory user. This behavior can be overridden as described in [“Running idsync resync” on page 52](#).
- Before a new entry can be synchronized to Directory Server or Active Directory, its DN is constructed using the creation expression, which is specific to each SUL.

The default behavior of `idsync resync` can be overridden if Example Bank wants to synchronize user creations from Directory Server to Active Directory, that is, the SUL_AD_WEST and SUL_AD_EAST can have the same base DN and filter.

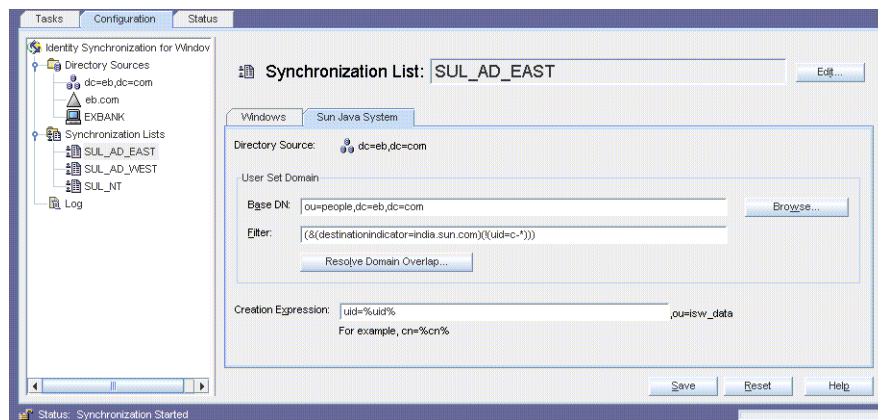
Thus, you can ignore the previous warning message and, as a workaround, rearrange the filters. Change the `(&(! (uid=c-*)) (destinationindicator=eb.com))` filter to be equivalent to `(&(destinationindicator=eb.com) (! (uid=c-*)))`. This rearrangement passes the console's validations because the console only performs the string comparisons to determine if the two filters are equivalent.

Note – If there is no SUL filter or it has only a single component, an SUL filter (or filter component) is added. This SUL filter component is different for each SUL but always evaluates to true.

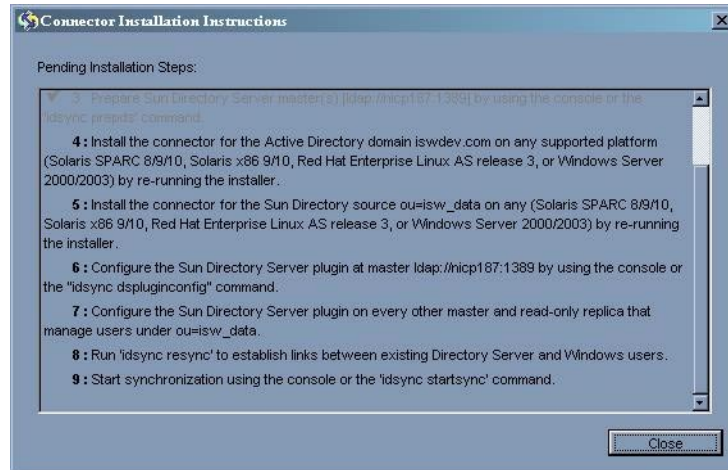
For example, the SUL definitions for SUL_AD_WEST and SUL_AD_EAST could also be defined as follows:

- (&(! (uid=c-*) (destinationindicator=eb.com) (!(bogusAttr=SUL_AD_WEST))))
- (&(! (uid=c-*) (destinationindicator=eb.com) (!(bogusAttr=SUL_AD_EAST))))

The new definition of SUL_AD_EAST passes the Console's validation.

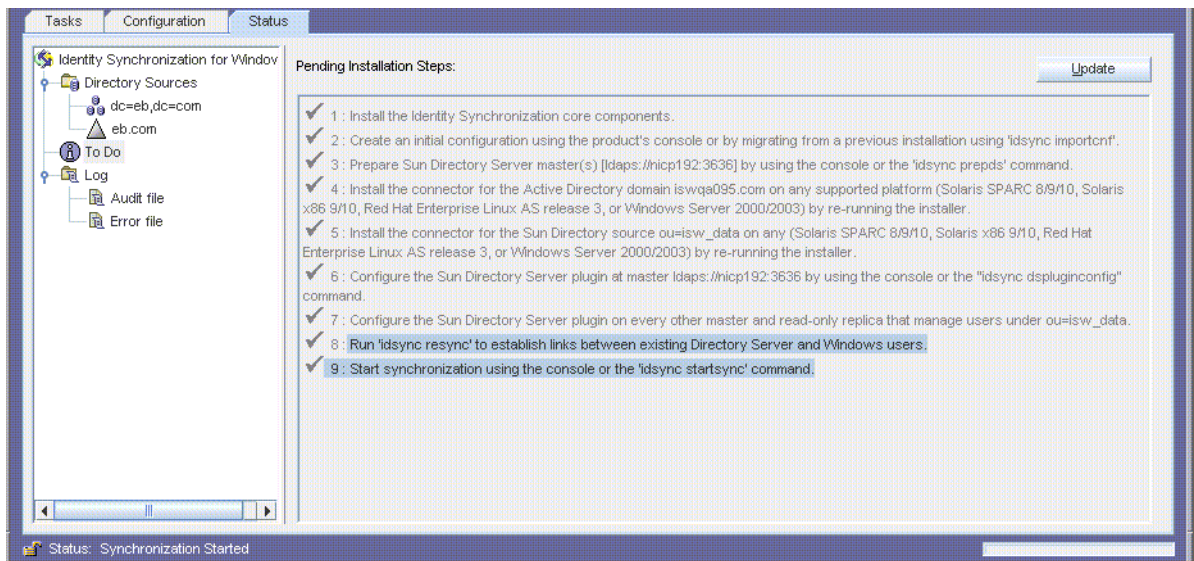


After the configuration is saved successfully, you install the connectors and the Directory Server plug-ins.



Installing the Connectors and Directory Server Plug-Ins

The Active Directory and Directory Server connectors are installed on master-east.eb.com. The Windows NT Connector is installed on the Primary Domain Controller pdc-east.eb.com. The Directory Server plug-ins are installed on the two preferred Directory Servers, that is, master-east.eb.com and master-west.eb.com.



Perform the `idsync resync` procedure to link all the existing users as there are no other preferred Directory Servers or read-only replicas in this environment.

Running `idsync resync`

In Example Bank's deployment, users already have accounts in Directory Server and in Windows. You must run `idsync resync` to establish links between equivalent users before starting synchronization. Use the `-f <linking-file> resync` option to link the users. For detailed information about the `idsync resync` command, see “Using `idsync resync`” in *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide*, in *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide*.

Running `idsync resync` initializes the `destinationindicator` attribute in each Directory Server entry, which ensures that the existing users in Directory Server match their SUL filter. If this attribute is not initialized with each user's Windows domain, changes to Directory Server users do not propagate back to Windows because the entry does not match the `destinationindicator` part of the SUL filter. In situations where the `destinationindicator` attribute is not populated, running the `idsync resync` command without the `-k` option establishes links between the users.

All users with Active Directory accounts have a Directory Server password that is synchronized with their Active Directory password, using the `-i NEW_LINKED_USERS` option.

For example, the process of linking a single user “John Test” is described here. This user has an Active Directory account in the `ou=east` organizational unit. The user entry is as follows:

```
bash-2.05# ./ldapsearch -h ad-west.eb.com -b "dc=eb,dc=com" -D
"cn=Administrator,cn=users,dc=eb,dc=com" -w <password omitted>
"samaccountname=jtest" version: 1
dn: CN=John Test,OU=east,DC=eb,DC=com
accountExpires: 9223372036854775807
badPasswordTime: 0
badPwdCount: 0
codePage: 0
cn: John Test
countryCode: 0
displayName: John Test
givenName: John
instanceType: 4
lastLogoff: 0
lastLogon: 0
logonCount: 0
distinguishedName: CN=John Test,OU=east,DC=eb,DC=com
objectCategory: CN=Person,CN=Schema,CN=Configuration,DC=eb,DC=com
objectClass: top
objectClass: person
```

```

objectClass: organizationalPerson
objectClass: user
objectGUID:: dYGjjEBYukyXXMJ//08KNw==
objectSid:: AQUAAAAAAAAUAAAAFSWvR+sleSxDFwoyUwQAAA==
primaryGroupID: 513
pwdLastSet: 127426694450768912
name: John Test
sAMAccountName: jtest
sAMAccountType: 805306368
sn: Test
userAccountControl: 512
userPrincipalName: jtest@eb.com
uSNChanged: 7043
uSNCreated: 7039
whenChanged: 20041019142405.0Z
whenCreated: 20041019142404.0Z

```

The user's password in Active Directory is abc:

```

bash-2.05# ./ldapsearch -h ad-west.eb.com
-b "dc=eb,dc=com"
-D "cn=John Test,ou=east,dc=eb,dc=com"
-w abc "samaccountname=jtest" version: 1
dn: CN=John Test,OU=east,DC=eb,DC=com
cn: John Test

```

The user's Directory Server account is as follows:

```

bash-2.05# ./ldapsearch -h master-east.eb.com
-b "dc=eb,dc=com" -D "cn=Directory Manager"
-w <password omitted> "uid=jtest" version: 1
dn: uid=jtest,ou=People, dc=eb,dc=com
uid: jtest
givenName: John
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
sn: Test
cn: John Test
userPassword: {SSHA}CwM7vTIJUEH+aahj1kUH/1/CruIKJ1Vw1hN0eA==

```

The user's password in Directory Server is 123:

```

bash-2.05# ./ldapsearch -h master-east.eb.com
-b "dc=eb,dc=com" -D "uid=jtest,ou=people,dc=eb,dc=com"
-w 123 "uid=jtest" cn version: 1
dn: uid=jtest,ou=People, dc=eb,dc=com
cn: John Test

```

The following file is used to link Active Directory users and Windows NT users to the equivalent Directory Server users.

For information on the syntax used in this example, see *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide*. Samples are available in the `samples/` directory where Core is installed.

A separate section is provided for each SUL. Active Directory and Directory Server users are linked if their Active Directory `samaccountname` attribute matches their Directory Server `uid` attribute. Windows NT and Directory Server users are linked if their Windows NT `USER_NAME` attribute matches their Directory Server `uid` attribute.

```
<?xml version="1.0" encoding="UTF-8"?>
<UserLinkingOperationList>
  <UserLinkingOperation parent.attr="UserLinkingOperation"
                        sulid="SUL_AD_EAST">
    <UserMatchingCriteria parent.attr="UserMatchingCriteria">
      <AttributeMap parent.attr="AttributeMap">
        <AttributeDescription parent.attr="SunAttribute" name="uid"/>
        <AttributeDescription parent.attr="WindowsAttribute"
                              name="samaccountname"/>
      </AttributeMap>
    </UserMatchingCriteria>
  </UserLinkingOperation>
  <UserLinkingOperation parent.attr="UserLinkingOperation"
                        sulid="SUL_AD_WEST">
    <UserMatchingCriteria parent.attr="UserMatchingCriteria">
      <AttributeMap parent.attr="AttributeMap">
        <AttributeDescription parent.attr="SunAttribute" name="uid"/>
        <AttributeDescription parent.attr="WindowsAttribute" name="samaccountname"/>
      </AttributeMap>
    </UserMatchingCriteria>
  </UserLinkingOperation>
  <UserLinkingOperation parent.attr="UserLinkingOperation" sulid="SUL_NT">
    <UserMatchingCriteria parent.attr="UserMatchingCriteria">
      <AttributeMap parent.attr="AttributeMap">
        <AttributeDescription parent.attr="SunAttribute" name="uid"/>
        <AttributeDescription parent.attr="WindowsAttribute" name="USER_NAME"/>
      </AttributeMap>
    </UserMatchingCriteria>
  </UserLinkingOperation>
</UserLinkingOperationList>
```

When the `idsync resync` command with the `linkusers.cfg` file is executed, a message that the `-i` option is not supported for Windows NT SULs is displayed:

```
bash-2.05# ./idsync resync -w <omitted password> -q
<omitted password> -f linkusers.cfg -i NEW_LINKED_USERS
```

Validating and starting refresh operation "1098189761942'.
Hit CTRL+C to cancel. The operation cannot be started because passwords cannot be reset from Windows NT Synchronization User Lists. The resync operation must not include the "SUL_NT' Synchronization User List. Please remove this option or explicitly specify non-Windows NT Synchronization User Lists using the -l option.

Split the linkusers.cfg file into a file that has only the Windows NT SUL, and the following file, linkusers-ad-only.cfg, that has both Active Directory SULs:

```
<?xml version="1.0" encoding="UTF-8"?>
<UserLinkingOperationList>
  <UserLinkingOperation parent.attr="UserLinkingOperation"
    sulid="SUL_AD_EAST">
    <UserMatchingCriteria parent.attr="UserMatchingCriteria">
      <AttributeMap parent.attr="AttributeMap">
        <AttributeDescription parent.attr="SunAttribute" name="uid"/>
        <AttributeDescription parent.attr="WindowsAttribute" name="samaccountname"/>
      </AttributeMap>
    </UserMatchingCriteria>
  </UserLinkingOperation>
  <UserLinkingOperation parent.attr="UserLinkingOperation"
    sulid="SUL_AD_WEST">
    <UserMatchingCriteria parent.attr="UserMatchingCriteria">
      <AttributeMap parent.attr="AttributeMap">
        <AttributeDescription parent.attr="SunAttribute" name="uid"/>
        <AttributeDescription parent.attr="WindowsAttribute" name="samaccountname"/>
      </AttributeMap>
    </UserMatchingCriteria>
  </UserLinkingOperation>
</UserLinkingOperationList>
```

The idsync resync command is run again by using the new linkusers-ad-only.cfg file and the -a option to run the command for the test user John Test. However, a message is displayed indicating that one entry matched a user in Directory Server, but the Directory Server user was not found in any SUL.

The Directory Server users do not have their destinationindicator attributes populated with the correct Windows domain names. Therefore, the test user did not match any of the SUL filters.

```
bash-2.05# ./idsync resync -w <omitted password>
-q <omitted password> -f linkusers-ad-only.cfg
-i NEW_LINKED_USERS -a "(samaccountname=jtest)"
Validating and starting refresh operation "1098193309618'.
Hit CTRL+C to cancel.
```

```
User progress:
# Entries sent: 1
User progress:
# Entries sent: 1
# Entries that matched a user that is in no SUL: 1
SUCCESS
```

To address this issue, the `allowLinkingOutOfScope="true"` parameter is added to the `linkusers-ad-only.cfg` file:

Note – Whenever a configuration has multiple SULs, use the `allowLinkingOutOfScope=true` parameter.

```
<?xml version="1.0" encoding="UTF-8"?>
<UserLinkingOperationList allowLinkingOutOfScope="true">
  <UserLinkingOperation parent.attr="UserLinkingOperation"
    sulid="SUL_AD_EAST">
    <UserMatchingCriteria parent.attr="UserMatchingCriteria">
      <AttributeMap parent.attr="AttributeMap">
        <AttributeDescription parent.attr="SunAttribute" name="uid"/>
        <AttributeDescription parent.attr="WindowsAttribute"
          name="samaccountname"/>
      </AttributeMap>
    </UserMatchingCriteria>
  </UserLinkingOperation>
  <UserLinkingOperation parent.attr="UserLinkingOperation" sulid="SUL_AD_WEST">
    <UserMatchingCriteria parent.attr="UserMatchingCriteria">
      <AttributeMap parent.attr="AttributeMap">
        <AttributeDescription parent.attr="SunAttribute" name="uid"/>
        <AttributeDescription parent.attr="WindowsAttribute" name="samaccountname"/>
      </AttributeMap>
    </UserMatchingCriteria>
  </UserLinkingOperation>
</UserLinkingOperationList>
```

When the `idsync resync` command is executed again, the test user is successfully linked and updated with the `destinationindicator` attribute value.

```
bash-2.05# ./idsync resync -w <omitted password>
-q <omitted password> -f linkusers-ad-only.cfg
-i NEW_LINKED_USERS -a "(samaccountname=jtest)"
Validating and starting refresh operation "1098191329451".
Hit CTRL+C to cancel.
User progress:
# Entries sent: 1
```



```
User progress:# Entries sent: 1
# Entries successfully linked: 1
# Entries that were modified: 1
SUCCESS
```

The following changes occur in the Directory Server entry:

- The `dspswuserlink` attribute is set to match the `objectguid` attribute in the Active Directory entry.
- The `destinationindicator` attribute is set to match the Active Directory domain name (`eb.com`).
- The `dspswvalidate` operation attribute is set to true, which implies that the Directory Server password is stale and the user requires on-demand password synchronization.
- The value does not change until the user logs in to Directory Server using the Active Directory password, triggering on-demand password synchronization. Although the Directory Server password was reset to the Active Directory value, the `userPassword` attribute has not changed.
- The user now has the `dspswuser` auxiliary object class, which allows the use of Identity Synchronization for Windows specific attributes in the user entry (for example, `dspswuserlink`).

```
bash-2.05# ./ldapsearch -h master-west.eb.com
-b "dc=eb,dc=com" -D "cn=Directory Manager"
-w <omitted password>\> "uid=jtest" "*"
dspswvalidate version: 1 dn: uid=jtest,ou=People, dc=eb,dc=com
dspswvalidate: true
dspswuserlink:: dYGjjEBYukyXXMJ//08KNw==
destinationindicator: eb.com
cn: John Test
userPassword: {SSHA}sTpxX8RQcz4GjqJ0ttSauXNjWcnaR/hC1X7gPA==
uid: jtest
givenName: John
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
objectClass: dspswuser
sn: Test
```

Verify that John Test can log in to Directory Server using the Active Directory password (abc):

```
bash-2.05# ./ldapsearch -h master-east.eb.com
-b "dc=eb,dc=com" -D "uid=jtest,ou=people,dc=eb,dc=com"
-w abc "uid=jtest" cn version: 1
dn: uid=jtest,ou=People, dc=eb,dc=com
cn: John Test
```

After the user has logged into Directory Server and when an `ldapsearch` is executed, the on-demand password synchronization has removed the `dspswvalidate` attribute and updated the `userPassword` attribute:

```
bash-2.05# ./ldapsearch -h master-west.eb.com
-b "dc=eb,dc=com" -D "cn=Directory Manager"
-w <omitted password> "uid=jtest" "*"
dspswvalidate version: 1
dn: uid=jtest,ou=People, dc=eb,dc=com
userPassword: {SSHA}8wmyeFe2bLr0kWm/SUStqmx63CeIHCASLFujUQ==
dspswuserlink:: dYGjjEBYukyXXMJ//08KNw==
destinationindicator: eb.com
cn: John Test
uid: jtest
givenName: John
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
objectClass: dspswuser
sn: Test
```

To link all of the Active Directory users, the same `idsync resync` command is executed without the `-a` option:

```
bash-2.05# ./idsync resync -w <omitted password>
-q <omitted password> -f linkusers-ad-only.cfg
-i NEW_LINKED_USERS
```

To link all of the Windows NT users, the `idsync resync` command is run on the `linkusers-nt-only.cfg` file, which contains information about `SUL_NT`: (without the `-i NEW_LINKED_USERS` option)

```
bash-2.05# ./idsync resync -w <omitted password>
-q <omitted password> -f linkusers-nt-only.cfg
```

When Example Bank links all of its users by running `idsync resync`, most of the users are linked successfully, but some users cannot be linked due to data inconsistencies. After these inconsistencies are manually corrected, `idsync resync` is run again to link the remaining users.

The next section discusses how to resolve an issue when users are migrated from Windows NT to Active Directory.

Running the Resynchronization Procedure When Directory Server Is Authoritative

When `idsync resync` is run without the `-k` option, which only links users, all synchronized attributes in the user entry are updated. In the previous examples in this overall section, the `destinationindicator` attribute is automatically populated with the correct Windows domain name. The `cn` and `uid` (Directory Server attributes) are also updated because they are synchronized.

The users are linked based on `uid`. The `uid` is already in sync, but the `cn` in Directory Server might be replaced with a value from Active Directory. This process might not be appropriate when Directory Server has the authority of these attributes.

▼ To Synchronize Attribute Values in Active Directory With the Values in Directory Server After Linking Entries

- 1 **Change the configuration so the only synchronized attributes are `userPassword` and `destinationindicator`.**
- 2 **Execute the `idsync resync -f <linking file>` command to link the entries and populate the `destinationindicator` attribute.**
- 3 **Change the configuration to include all the synchronized attributes. For example, add `cn` and `uid`.**
- 4 **Execute the `idsync resync -o Sun` command to synchronize the Active Directory attributes with their Directory Server values.**

Note – If the `destinationindicator` attribute does not need to be populated, execute the `idsync resync -f <linking file> -k` command to only link the entries. Then execute the `idsync resync -o Sun` command to synchronize the Directory Server attribute values from Directory Server to Active Directory.

Configuration and Installation Summary

This section summarizes the configuration tasks based on the main requirements of Example Bank.

Multiple Domains

Configuring Identity Synchronization for Windows to support multiple domains involves the following:

- Setting up `destinationindicator <-\> activedirectorydomainname <-\> user_nt_domain_name` as a synchronized attribute
- Using `destinationindicator` in the SUL filters so that entries modified in Directory Server can be located in the proper Active Directory domain
- When linking users by using the `idsync resync` command, specifying `allowLinkingOutOfScope="true"` in the input file.

Do not specify the `-k` option because you want the `destinationindicator` attribute to be primed.

PAM LDAP

Configuring Identity Synchronization for Windows to support PAM LDAP involves the following:

- Adding `shadowAccount` as an auxiliary object class for Directory Server
- Adding the creation attribute default values for various `shadowAccount` attributes

For information about the prerequisites and how to conform to PAM LDAP on the Solaris OS, see [Appendix A](#).

WAN Deployment

Identity Synchronization for Windows has limited support for WAN deployments and can be synchronized with the Directory Server or Active Directory domain controllers that are only available over the WAN. However, the Identity Synchronization for Windows Core and all the connectors must be installed on the same LAN.

The setup in this scenario was achieved by installing the following:

- Identity Synchronization for Windows Core.
- Directory Server Connector.
- Active Directory Connector on the same machine where Identity Synchronization for Windows Core and Directory Server Connector are installed.
- Windows NT Connector on a machine in the same LAN.

In this case study, the Active Directory Connector communicates across the WAN with the Active Directory domain controller on the west coast. A domain controller is available on the east coast, but because it is not the PDC FSMO role owner, synchronization would be significantly delayed if it was selected.

Note – When the Directory Server domain controller and Active Directory domain controller are separated by a WAN, you have the option of installing Identity Synchronization for Windows in one of the following:

- On the same LAN as Directory Server
- On the same LAN as Active Directory
- Somewhere in between

In general, the best performance is achieved when Identity Synchronization for Windows is installed on the same LAN as Directory Server.

Identity Synchronization for Windows has been tested in a variety of WAN environments, but it requires minimum a link with at least T1 (1.44 Mb/sec) speeds and a round-trip latency of no more than 300 milliseconds.

Migrating Users From Windows NT to Active Directory

This section explains how to migrate users between Windows domains, which involves the following:

- Moving user accounts from Windows NT to Active Directory
- Unlinking migrated Windows NT entries
- Linking migrated Active Directory entries

Example Bank wants to migrate users each week from Windows NT to Active Directory based on each user's last name. Example Bank has already migrated users whose last name starts with A-F.

After the accounts have been migrated to Active Directory, changes made to migrated users in Directory Server or Active Directory will not be synchronized:

- Changes made in Directory Server are detected. However, each entry will still be linked to a Windows NT entry. The entry is processed by the Windows NT Connector, but the user no longer exists in Windows NT.
- Changes made in Active Directory are also detected but the entry does not match a Directory Server entry because it has not been linked.

To establish links between the migrated Active Directory accounts and the Directory Server entries, the links are removed from the migrated Windows NT accounts. Then, the Active Directory accounts are linked to their corresponding Directory Server entries.

Note – To avoid losing too many changes, perform the full migration when the load on Directory Server is light. The `idsync resync` command can be run to recover any changes except changes to the passwords.

In this scenario, the users are moved from a Windows NT domain to an Active Directory domain, but the same procedure can be followed when moving users are between Windows NT domains or Active Directory domains.

Unlinking Migrated Windows NT Entries

To unlink migrated users, the `dspswuserlink` attribute is removed from each Directory Server entry that is affected. The following sample script can be used to remove this attribute from a large number of users. The script accepts `ldapsearch` arguments for the users that should be unlinked.

```
#!/usr/bin/perl
# This script is used to break the link between Directory Server
# entries and the corresponding Windows entries. Provide
# complete ldapsearch arguments for the users to unlink.
# If many users are unlinked, use -D "cn=Directory Manager" to
# avoid search results limits imposed by the directory server.
# ldapsearch and ldapmodify must be in the path, and the
# current directory must be writable.
#
# Modify these variables to point to the ldapsearch and
# ldapmodify commands that ship with the Sun Directory Server.
# The versions that ship with Solaris will not work in this script.
die "Edit this script to modify these variables and then remove this
line.\n";
my $LDAPSEARCH_EXE = "/opt/mps/serverroot/shared/bin/ldapsearch";
my $LDAPMODIFY_EXE = "/opt/mps/serverroot/shared/bin/ldapmodify";
my $USAGE = "Usage: unlink.pl <ldapsearch args for users to unlink>\n";
# Valid ldapsearch options that don't apply to ldapmodify
my %INVALID_LDAPMODIFY_OPTS = ("-b" => 1, "-s" => 1);
# Valid ldapsearch options that have arguments and don't apply
# to ldapmodify
my %INVALID_LDAPMODIFY_OPTS_WITH_ARG = ("-b" => 1, "-s" => 1);
# The file name for the file to hold the unlink ldif
my $UNLINK_LDIF_FILE = "unlink.ldif";
#
# SCRIPT BEGIN
#
scalar(@ARGV) \> 0 or die "$USAGE";
# Run ldapsearch to find the users to unlink.
my $ldapsearchArgs = getLdapsearchArgs(@ARGV);
my $ldapsearchCmd = "$LDAPSEARCH_EXE $ldapsearchArgs";
```

```

my $matchedUsersLdif = "$ldapsearchCmd";
lastCommandSucceeded() or die "Failed when running ". "$ldapsearchCmd.\n$USAGE";
# Construct ldif to unlink each matched user.
my @userDns = parseDnsFromLdif($matchedUsersLdif);
print "Matched ".scalar(@userDns)." linked entries.\n";
my $unlinkLdif = constructUnlinkLdif(@userDns); my $fileName =
writeLdifToFile($unlinkLdif);
# Run ldapmodify to unlink the users.
my $ldapmodifyArgs = getLdapmodifyArgs($fileName, @ARGV);
my $ldapmodifyCmd = "$LDAPMODIFY_EXE $ldapmodifyArgs";
"$ldapmodifyCmd";
print "Unlinked ".scalar(@userDns)." entries.\n";
lastCommandSucceeded() or die "Failed when running ". "$ldapmodifyCmd.\n$USAGE";
exit 0;
#
# SCRIPT END
# Return true iff the last command succeeded.
#
sub lastCommandSucceeded {
    return ($? \>\> 8) == 0;
}
#
# Return the dns in the ldif.
#
sub parseDnsFromLdif { my $ldif = shift @_ ; my @dns = ();
# Note that DNSs can span multiple lines.
while ($ldif =~ /(^\dn:.*(\\n^[ ]+\\S+.*))/gmi) {
    push @dns, $1;
}
return @dns;
}
#
# Return ldif for all users to unlink
#
sub constructUnlinkLdif {
    my $ldif = "";
    for my $dn (@_) {
        $ldif .=
            "$dn\n" .
            "changetype: modify\n" .
            "replace: dspswuserlink\n" .
            "-\n" .
            "\n";
    }
    return $ldif;
}
#
# Writes ldif to a file and returns the name of the file.

```

```
#
sub writeLdifToFile {
    my $ldif = shift @_;
    open(LDIF, ">$UNLINK_LDIF_FILE") or
        die "Could not open $UNLINK_LDIF_FILE for writing.\n";
    print LDIF $ldif;
    close(LDIF);
    return $UNLINK_LDIF_FILE;
}
#
# Return the arguments to use for ldapsearch as a single string
#
sub getLdapsearchArgs {
    # Always use -L because Solaris's ldapsearch doesn't
    # return ldif by default.
    my $ldapsearchArgs = "";
    # Modify the last argument to include the search filter
    my $lastIndex = $#;
    $_[$lastIndex] = getLinkedSearchFilter($_[$lastIndex]);
    for my $arg (@_) {
        $ldapsearchArgs .= " '$arg'";
    }
    return $ldapsearchArgs;
}
#
# Construct an ldapfilter that only matches linked users.
#
sub getLinkedSearchFilter {
    my $filter = shift @_;
    if (!$filter =~ /^</>) {
        $filter = "($filter)";
    }
    return "(&(dpswuserlink=*)$filter)";
}
#
# Return the arguments to use for ldapmodify as a single string.
#
sub getLdapmodifyArgs {
    my $ldifFile = shift @_;
    my $ldapmodifyArgs = "-f '$ldifFile'";
    my $prevArg = "";
    # Iterate through all args, skipping ones that don't
    # apply and the last one.
    for my $i (0..($# - 1)) {
        my $arg = $_[$i];
        if (!$INVALID_LDAPMODIFY_OPTS{$arg} &&
            !$INVALID_LDAPMODIFY_OPTS_WITH_ARG{$prevArg}) {
            $ldapmodifyArgs .= " '$arg'";
        }
    }
}
```



```

    }
    $prevArg = $arg;
  }
  return $ldapmodifyArgs;
}

```

The next batch of Example Bank users to be migrated from Windows NT to Active Directory have a last name that starts with the letter G. After these users have been migrated, the following `unlink.pl` script is executed to unlink these entries in Directory Server.

```

bash-2.05# ./unlink.pl -h master-east.eb.com
-b "dc=eb,dc=com" -D "cn=Directory Manager"
-w <omitted password> " (sn=G*)"
Matched 1346 linked entries.
Unlinked 1346 entries.

```

In this example, 1346 entries were migrated and thus unlinked. The Directory Manager dn is provided to avoid search results limits in the Directory Server. If other accounts are used, make sure that their search capabilities are not limited, to avoid unlinking only subsets of users.

Linking Migrated Active Directory Entries

After the links in the Directory Server entries are removed, new links are established with the Active Directory entries by using the `idsync resync` command. Use the `-a` option with the `(sn=G*)` filter to link only the users that have been migrated.

According to Microsoft's documentation, user passwords will be migrated when users are moved from Windows NT to Active Directory. However, if users change their passwords in Active Directory before they are relinked to the Directory Server entries, these password changes will not be synchronized to the Directory Server.

You can use the `-i NEW_LINKED_USERS` option with the `idsync resync` command to synchronize Directory Server passwords with their Active Directory values.

Note – If any of the users' passwords are modified in Directory Server during the migration process, these password changes will be lost.

```

bash-2.05# ./idsync resync -w <omitted password>
-q <omitted password> -f linkusers-ad-only.cfg
-i NEW_LINKED_USERS -a "(sn=G*)"
Validating and starting refresh operation '1098238348483'.
Hit CTRL+C to cancel.
User progress:
# Entries sent: 1346
# Entries successfully linked: 1346

```

```
# Entries that were modified: 1346
SUCCESS
```

Moving Users Between Active Directory Organizational Units

When a user is moved from one Active Directory container to another (for example, from ou=west to ou=east), no action needs to be taken for this user to continue to be synchronized by Identity Synchronization for Windows.

When Contractors Become Full-Time Employees

When a contractor becomes a full-time employee, the special c - prefix is removed from the person's login name. The new full-time employee is now in SUL for the first time, and the entry will be interpreted as being new even though it was not recently created. If the contractor has an Active Directory entry that is modified, Identity Synchronization for Windows will attempt to create the entry in Directory Server.

The following table provides the guidelines for handling contractor accounts when they become full-time employees.

TABLE 2-3 Guidelines for Transitioning Contractor to Employee Accounts

Active Directory Account	Directory Server Account	Creating Linked Entries in Active Directory and Directory Server
No account	No account	This kind of situation should not occur because contractors have either an Active Directory or Directory Server account. If it does occur, create a new entry in Active Directory, and Identity Synchronization for Windows automatically creates a new user in Directory Server.
No account	Account	<div>1. Remove the c - prefix from the Directory Server entry's uid.</div> <div>2. Create a new entry in Active Directory for the new full-time employee.</div> <div>3. Run idsync resync to establish a link for the new full-time employee. Use the -a option to limit the scope of the resync command to a single user.</div> <div>If a contractor's Directory Server entry is not important, do the following:</div> <div>1. Delete the Directory Server entry for the contractor, if there is one.</div> <div>2. Create a new entry in Active Directory.</div> <div>3. Identity Synchronization for Windows will create the corresponding new user in Directory Server.</div>
Account	No account	<div>Remove the c - prefix from the Active Directory entry's samaccountname.</div> <div>Identity Synchronization for Windows will interpret the change as a new user and create the corresponding new user in Directory Server.</div>

TABLE 2-3 Guidelines for Transitioning Contractor to Employee Accounts (Continued)

Active Directory Account	Directory Server Account	Creating Linked Entries in Active Directory and Directory Server
Account	Account	<div>1. Remove the c - prefix from the Directory Server entry's uid.</div> <div>2. Remove the c - prefix from the Active Directory entry's uid.</div> <div>Note – If this entry is modified before the Directory Server entry, the contractor will have two Directory Server accounts (the original one and a new one with a uid without the c - prefix)</div> <div>3. Run idsync resync to establish a link for the new full-time employee. Use the -a option to limit the scope of the resync command to a single user. If a contractor's Directory Server entry is not important, do the following:<div><div>1. Delete the Directory Server entry for the contractor, if there is one.</div><div>2. Remove the -c prefix from the Active Directory entry's samaccountname.</div><div>3. Identity Synchronization for Windows will create the corresponding new user in Directory Server.</div></div></div>

Case Study: Deploying in a High-Availability Environment Over a WAN Using SSL

The case study provided in this chapter explains how the company “Global Telco” implemented Sun Java System Identity Synchronization for Windows in a four-way multimaster replication (MMR) environment over a Wide Area Network (WAN) using the Secure Socket Layer (SSL).

This chapter covers the following topics:

- “Global Telco Deployment Information” on page 69
- “Installation and Configuration Overview” on page 74
- “Configuration Walkthrough” on page 80
- “Failover Installation” on page 88
- “Setting Up SSL” on page 91
- “Increasing Connector Worker Threads” on page 97
- “Aligning Primary and Failover Configurations” on page 98
- “Initial idsync resync Operations” on page 100
- “Periodic idsync resync Operations” on page 104
- “Configuring Identity Manager” on page 109
- “Understanding the Failover Process” on page 109
- “Initializing the Connector State” on page 111
- “Failover Installation Maintenance” on page 112
- “When to Fail over” on page 112
- “Failing Over” on page 113

Global Telco Deployment Information

This section describes Global Telco's existing architecture and what the company wants to achieve in this deployment. This section also lists the Identity Synchronization for Windows features that are highlighted in this case study.

Global Telco's Existing Architecture

Global Telco, a large company with 500,000 employees world-wide, is using Sun Java System Identity Manager (Identity Manager) to support users between Active Directory, Directory Server, Oracle RDBMS, Novel NDS, and other systems. The company has two main data centers: one in the United States, and one in Europe.

The company has a single Active Directory domain (gt.com) with four domain controllers, and a Sun Java System Directory Server deployment (dc=gt,dc=com) with four preferred Directory Servers and four read-only replicas.

Directory Server Information

The Sun Java System Directory Server topology includes four preferred Directory Server and four master replicas. Directory Server is the corporate directory server used to control access to web-based applications. The directory server has a single root suffix, dc=gt,dc=com. Information about users is stored in the ou=people, dc=gt,dc=example,dc=com container with uid as the naming attribute.

- Two preferred Directory Server and two master replicas are located in the United States (a separate configuration directory in the United States stores configuration information for these systems).
- Two preferred Directory Server and two master replicas are located in Europe (a separate configuration directory in Europe stores configuration information for these systems).

All four preferred Directory Server have replication agreements with each other, but the four master replicas only have replication agreements with two of the masters.

Note – Identity Synchronization for Windows treats hub replicas the same as read-only replicas. In many scenarios, using a hub replica is preferred to using a read-only replica because a hub can be easily promoted to a preferred Directory Server.

Active Directory Information

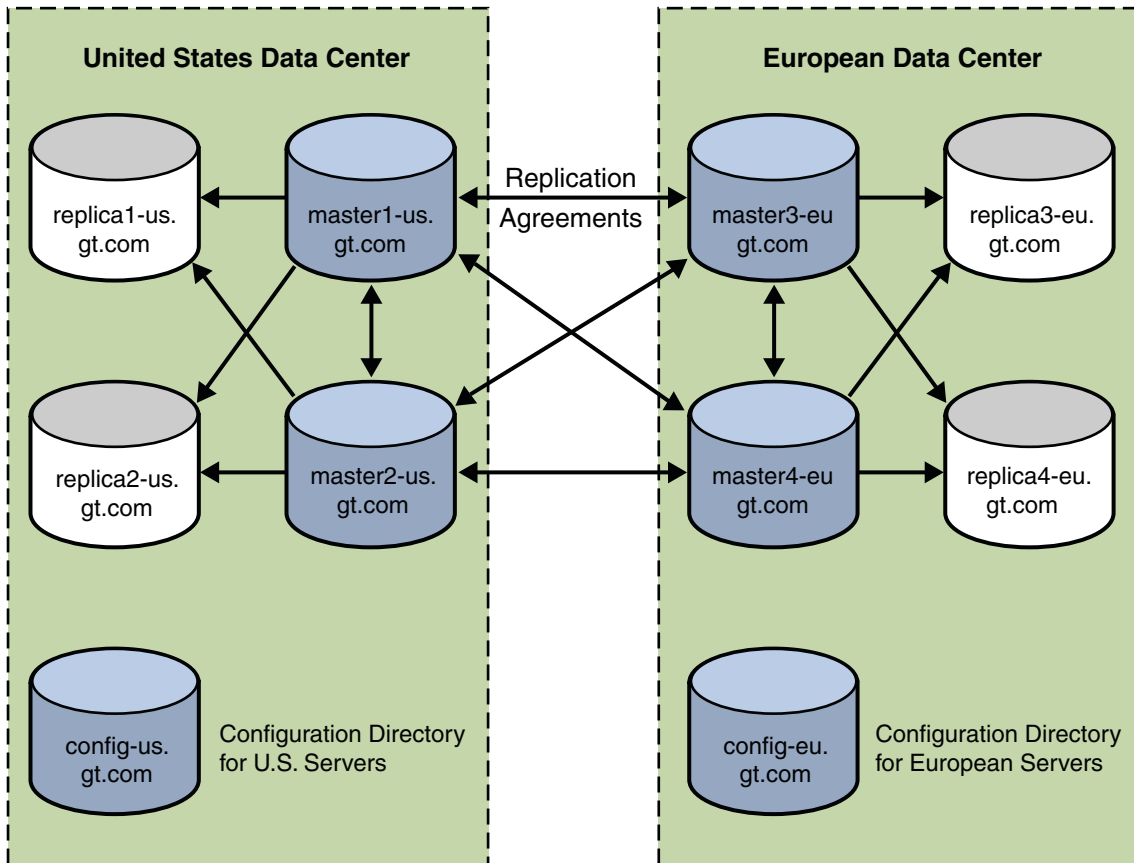


FIGURE 3-1 Data Center Information for Directory Server

The Active Directory deployment has a single domain, `gt.com`, with two domain controllers located in the United States and two in Europe. The user information is stored in the standard `cn=users` container in Active Directory (`cn=users,dc=gt,dc=com`).

The Active Directory `samaccountname` attribute value matches the Directory Server `uid` attribute. The Active Directory domain controller with the PDC FSMO role is located in the United States office.

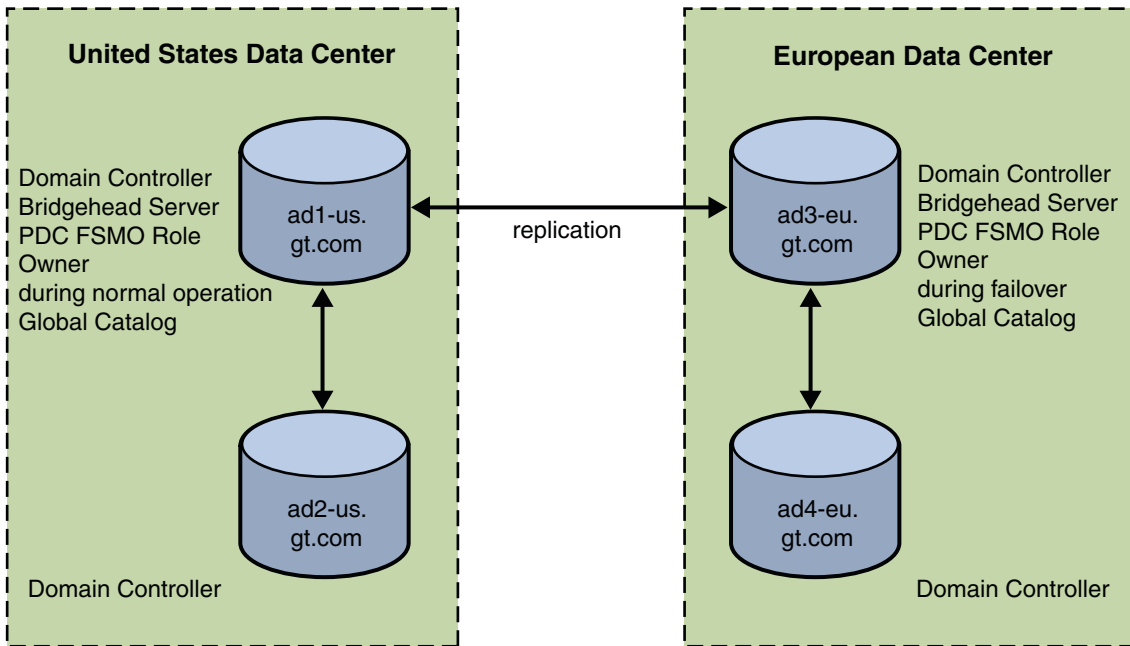


FIGURE 3-2 Data Center Information for Active Directory

Note – Both `ad1-us.gt.com` and `ad3-eu.gt.com` are bridgehead servers that control replication between the two sites.

Global Telco's Technical Requirements

Global Telco wants to achieve the following:

- Users' passwords for Windows systems must be synchronized with their Directory Server passwords.
- Users must be able to change passwords using native mechanisms made in either system, through the Change Password option in the Task Manager dialog box on Windows systems, and through a web-based portal for Directory Server.

Identity Synchronization for Windows supports capturing native password changes in Directory Server and Active Directory. Users can continue to change passwords as they always have.

Note – Passwords can be set in Directory Server by passing a pre-hashed password value. However, Identity Synchronization for Windows cannot synchronize passwords from Directory Server to Windows if the password is pre-hashed. Even in installations without Identity Synchronization for Windows, avoid using a pre-hashed password value because it circumvents password policy and password history.

- Existing Identity Manager functionality must be retained and continue to support users on Active Directory and Directory Server.

Identity Synchronization for Windows requires the users' Directory Server accounts to be explicitly linked to their Windows accounts. This linking is automatically done when Identity Synchronization for Windows is configured to synchronize creations of new users. However, because Identity Manager is provisioning both Active Directory and Directory Server accounts, Identity Synchronization for Windows will not synchronize new users. Global Telco must either run the `idsync resync` command periodically to link newly created users, or Identity Manager must be configured to set the necessary linking attributes when a new Directory Server entry is created.

- Support must be added for propagating native password changes made in Directory Server to all systems managed by Identity Manager.

Identity Manager supports synchronizing Active Directory password changes to many other systems because Identity Synchronization for Windows can synchronize password changes from Directory Server to Active Directory. Integrate Identity Manager with Identity Synchronization for Windows to synchronize password changes made in Directory Server to any system that Identity Manager supports.

- High availability for failover redundancy of all services is required in the European office.

Identity Synchronization for Windows is very robust. After all components are running, it synchronizes data without losing changes. By default, Identity Synchronization for Windows provides some high availability options, such as failover to a secondary Directory Server, and performing on-demand password synchronization against any Active Directory domain controller. It also includes a Watchdog that restarts failed processes.

However, if the machine that runs Identity Synchronization for Windows Core or Connector experiences a hardware failure, Identity Synchronization for Windows will not synchronize users until it is reinstalled on different hardware.

This case study addresses Global Telco's HA requirement by installing a completely separate instance of Identity Synchronization for Windows at the European office.

- All communication must use SSL and trusted certificates where possible.

Identity Synchronization for Windows supports SSL communication for all over-the-wire communication. By default, it does not require trusted certificates for SSL communication between connectors and directory sources, but it can be configured to require trusted certificates.

Identity Synchronization for Windows Features in This Case Study

The following Identity Synchronization for Windows features are used in this case study:

- Integrating Identity Synchronization for Windows with Identity Manager to synchronize passwords
- Deploying Identity Synchronization for Windows in an HA environment with a large number of users
- Installing Identity Synchronization for Windows on multiple machines
- Establishing links between users when Identity Synchronization for Windows does not synchronize account creation
- Providing directory server access over a WAN
- Using SSL

Installation and Configuration Overview

This section provides an overview of the installation and configuration tasks for meeting Global Telco's requirements. For details on configuring Identity Manager to coexist with Identity Synchronization for Windows, see [Appendix B](#).

Primary and Failover Installations

To provide a high-availability solution, Identity Synchronization for Windows must be installed in two separate locations, one in the United States and another in Europe. The deployment in the United States is the primary deployment, while the one in Europe is only meant to be used during failover scenarios.

To improve performance, the Identity Synchronization for Windows components are distributed between two machines in each location. For the deployment in the United States, the Identity Synchronization for Windows Core components are installed on `config-us.gt.com`, and both connectors are installed on `connectors-us.gt.com`. For the deployment in Europe, the Identity Synchronization for Windows Core components are installed on `config-eu.gt.com`, and both connectors are installed on `connectors-eu.gt.com`.

The primary deployment and the various communication paths are shown in the following figure. For simplicity, `gt.com` is dropped, and only the machine names are shown.

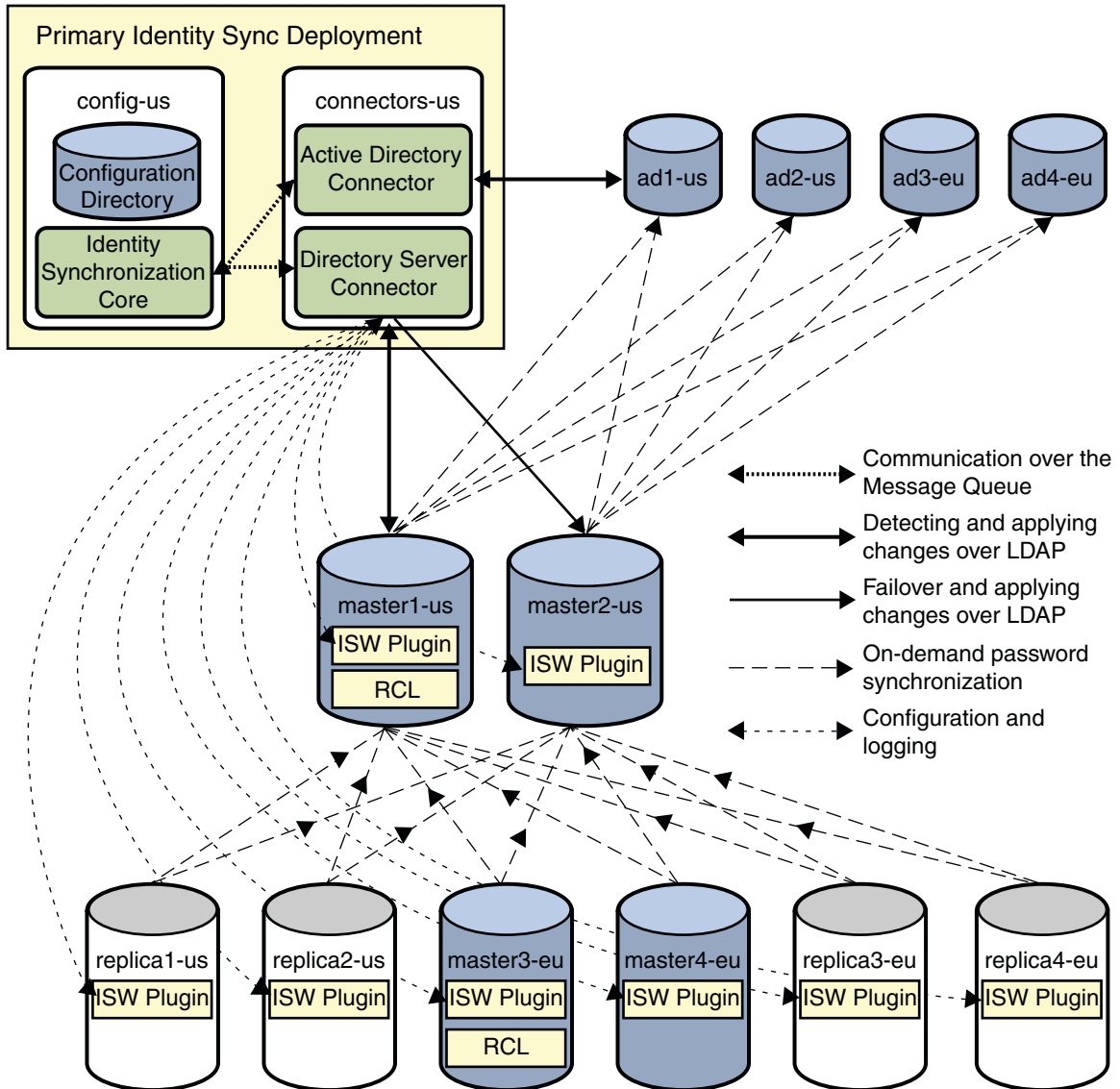


FIGURE 3-3 Primary Installation of Identity Synchronization for Windows

The Directory Server Connector and Active Directory Connector, installed on `connectors-us.gt.com`, communicate with each other and receive their configuration from the Message Queue that is installed with the Identity Synchronization for Windows Core.

The Active Directory Connector communicates exclusively with the `ad1-us.gt.com` domain controller, using LDAP. The Directory Server Connector communicates with two preferred

Directory Servers. While primary Directory Server is available, the Directory Server Connector detects and propagates changes to `master1-us.gt.com`. If primary Directory Server is unavailable, it fails over to `master2-us.gt.com` to apply changes, but cannot detect further changes made at any other preferred Directory Server until `master1-us.gt.com` is available.

Note –

- If both `master1-us.gt.com` and `master2-us.gt.com` are unavailable, changes synchronized from Active Directory are not applied to the other preferred Directory Servers, `master3-eu.gt.com` and `master4-eu.gt.com`. Identity Synchronization for Windows treats these other preferred Directory Servers like read-only replicas, except that external changes to these preferred Directory Servers are replicated to the retro change log on `master1-us.gt.com` and then synchronized to Active Directory.
 - The Identity Synchronization for Windows Directory Server Plug-ins running on `Master1-us.gt.com` and `master2-us.gt.com` only communicate with `ad2-us`, `ad3-eu`, and `ad4-eu` if `ad1-us` is unavailable. Similarly, the other preferred Directory Servers and replicas only communicate with `master2-us.gt.com` if `master1-us.gt.com` is unavailable.
-

An Identity Synchronization for Windows Directory Server Plug-in must be enabled on all eight Directory Server instances, four preferred Directory Servers and four read-only replicas.

You can enable the Directory Server Plug-in by using the following:

```
idsync dspluginconfig -{C/U} -D <bind DN> -w <bind pass word | -> [-h <CD  
hostname>] [-p <CD port no>] [-s <configuration suffix>] [-Z] [-P <cert db path>] [-m  
<secmod db path>] [-d <ds plugin hostname>] [-r <ds plug in port>] [-u <ds plugin user>]  
[-x <ds plugin user password>] [-o <database suf fix>] [-q <configuration password | ->]  
[-W] [-B <plugin DS cert db path>] [-g <plugin DS secmod db path>]
```

Type **idsync --help** for information about the syntax.

When a directory server starts, the Directory Server Plug-in establishes a secure connection to the Directory Server Connector. After the plug-in is authenticated, the connector sends the configuration information, and the plug-in can send log messages to the central log, through the connector. The configuration includes keys for encrypting modified passwords and Active Directory information for performing on-demand password synchronization.

When a user's Active Directory password changes, Identity Synchronization for Windows sets the `dspswvalide` attribute to `true` in the user's Directory Server entry. The user can then log in to any Directory Server, and on-demand password synchronization can originate from any server.

If the user logs in to `master1-us.gt.com` or `master2-us.gt.com`, on-demand password synchronization is performed directly to the `ad1-us.gt.com` Active Directory domain controller. Other domain controllers are contacted only if `ad1-us.gt.com` is unavailable.

If the user logs in to one of the other two preferred Directory Servers or one of the read-only replicas, on-demand password synchronization is performed against `master1-us.gt.com` or `master2-us.gt.com`. These preferred Directory Servers in turn continue the on-demand password synchronization to one of the Active Directory domain controllers.

These two actions are necessary for the following reasons:

- A read-only replica cannot update the user's Directory Server entry with the correct password if on-demand password synchronization succeeds.
- With the exception of the preferred and secondary Directory Servers, Identity Synchronization for Windows treats all Directory Server instances as read-only replicas.

After the primary installation is complete, the failover Identity Synchronization for Windows installation is performed on the two machines in Europe, `config-eu.gt.com` and `connectors-eu.gt.com`.

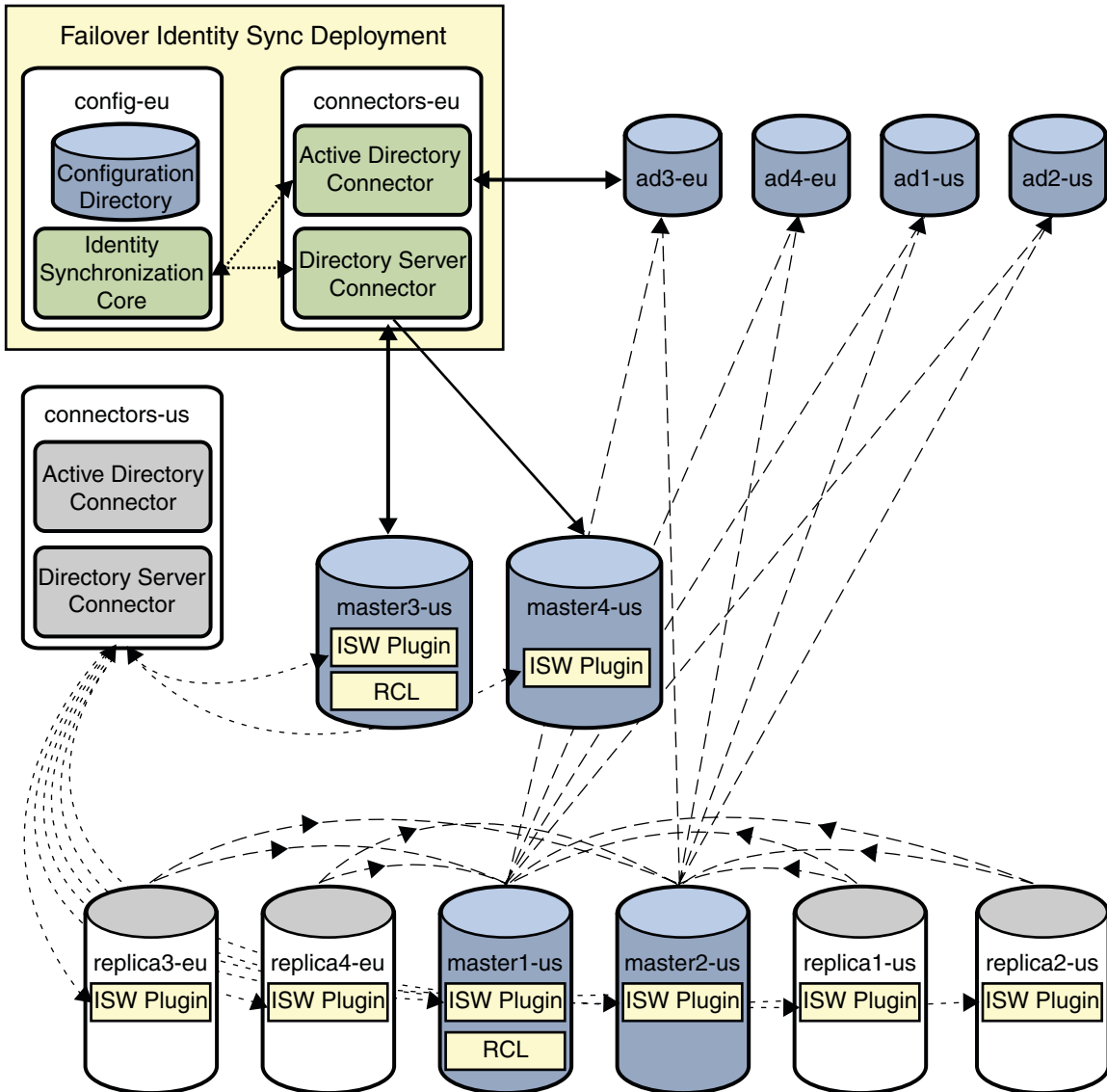


FIGURE 3-4 Failover Installation While the Primary Installation Is Active

Because the Identity Synchronization for Windows Directory Server Plug-ins have not been reinstalled, so they still receive their configuration from the Directory Server Connector running on `connectors-us.gt.com`, while on-demand password synchronization passes through `master1-us.gt.com` or `master2-us.gt.com` before reaching the Active Directory domain controllers.

The failover installation remains in this state until Global Telco needs to fail over to it. To complete the failover process, the Identity Synchronization for Windows Plug-in is enabled on every directory server, which changes its startup configuration to communicate with the Directory Server Connector running on connectors-eu.gt.com.

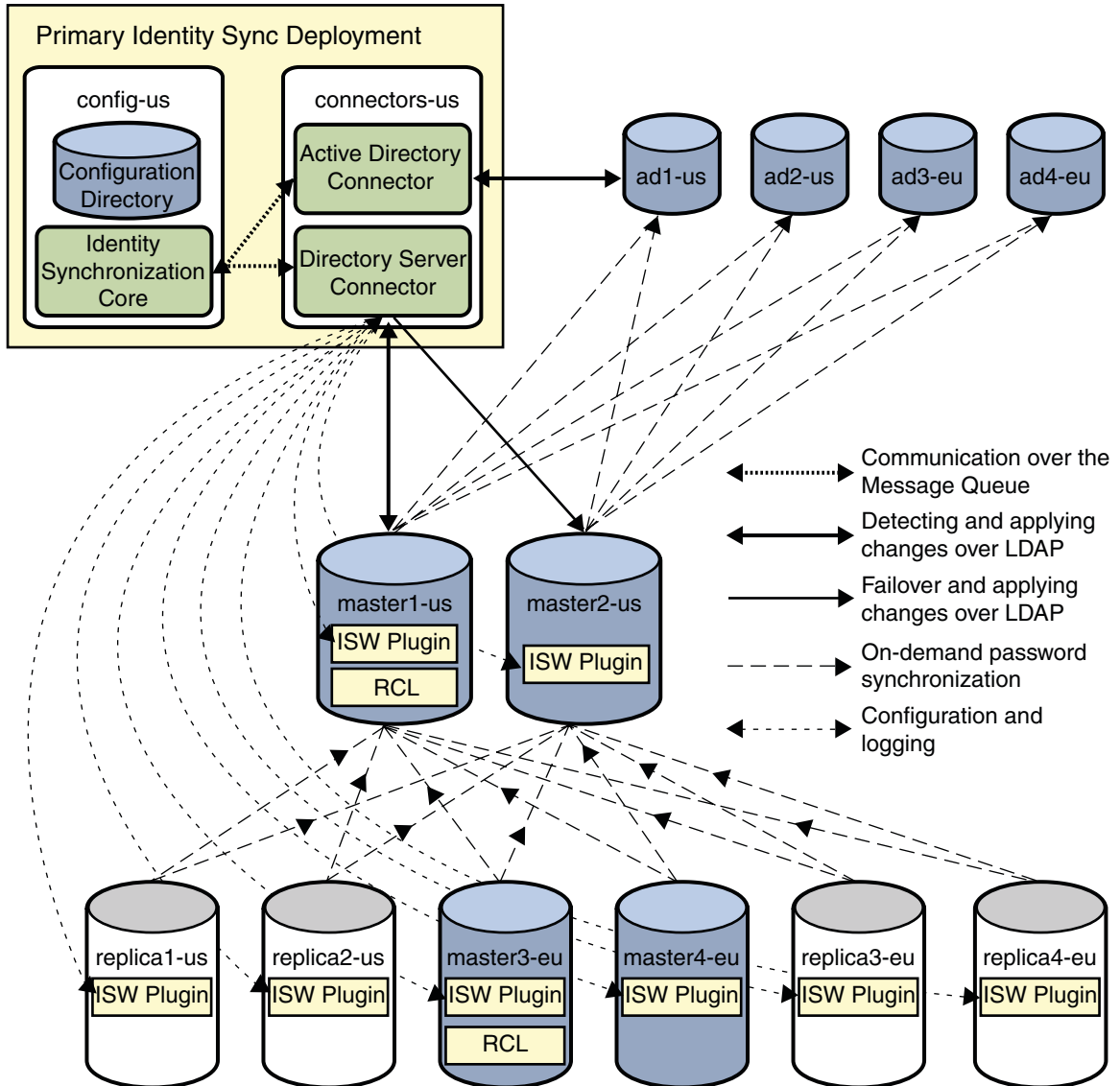


FIGURE 3-5 Primary Installation After Reinstalling the Identity Synchronization for Windows Plug-ins

Note – Setting up the failover installation significantly increases the amount of time required to deploy Identity Synchronization for Windows. However, this up-front cost is offset by having the ability to quickly fail over to the alternate deployment if necessary.

Periodically Linking New Users

Identity Synchronization for Windows is not used to synchronize the creation of new users. Therefore, the `idsync resync` command is executed periodically to establish links between newly created users. An LDAP filter is passed to this command to resynchronize only the subset of users that have been created since the command was last executed.

See [“Periodic `idsync resync` Operations” on page 104](#) for more information.

Large Deployment Considerations

Due to the large size of its deployment, Global Telco takes the following steps to increase the performance of Identity Synchronization for Windows.

- Identity Synchronization for Windows components are distributed across two dedicated machines.
- The log level is set to `INFO` except when diagnosing transient problems.
- Message Queue Broker memory limits are increased (see the *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide* for more information).
- The default number of worker threads in the connectors is increased.
- The `idsync resync` command is run over batches of users to reduce the peak load on Message Queue.

Each of these actions is discussed in detail in the next section.

Configuration Walkthrough

This section provides high-level steps to configure Identity Synchronization for Windows in a high-availability environment.

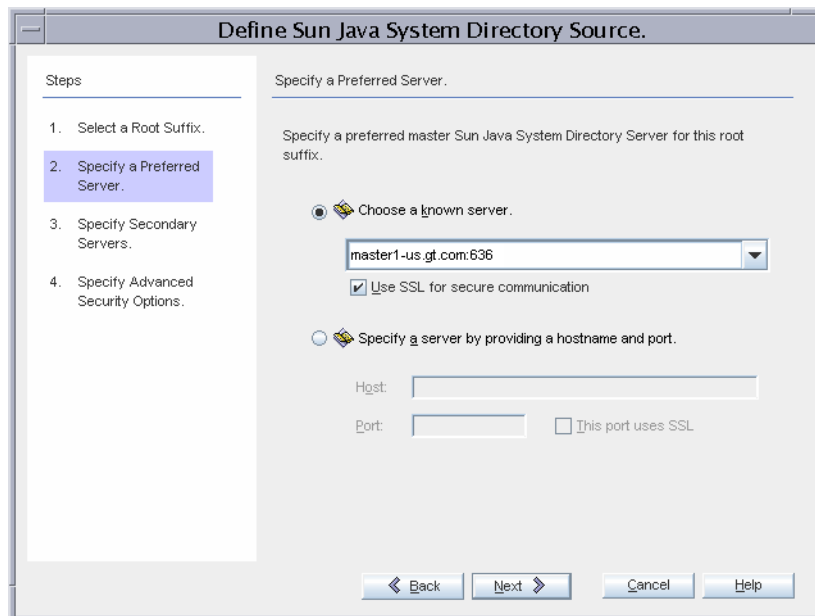
Note – Only important steps are provided. Any configuration instructions already discussed in the Example Bank case study have been omitted.

For detailed configuration instructions, see the *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide*.

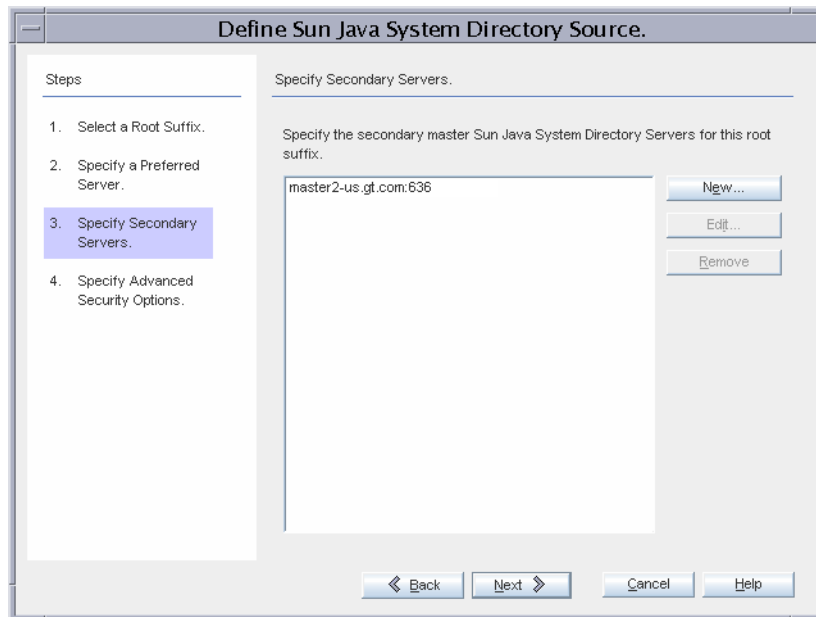
Primary Installation

After the Identity Synchronization for Windows Core is installed on `config-us.gt.com`, the Identity Synchronization for Windows Console is started. You configure the preferred Directory Server source first.

In this case, `master1-us.gt.com` is chosen as the preferred Directory Server. The connector communicates with the Directory Server source over SSL.

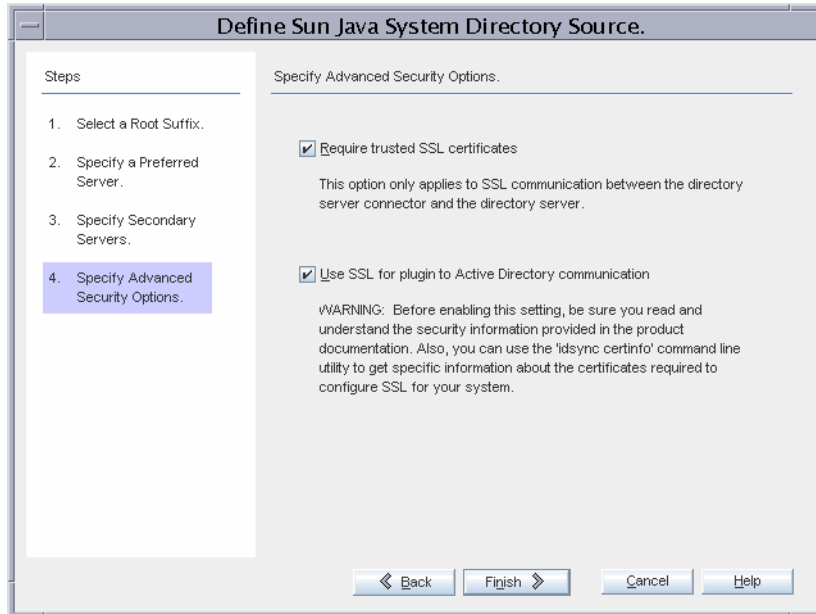


`master2-us.gt.com` is chosen as the secondary Directory Server. The connector communicates with Directory Server over SSL.

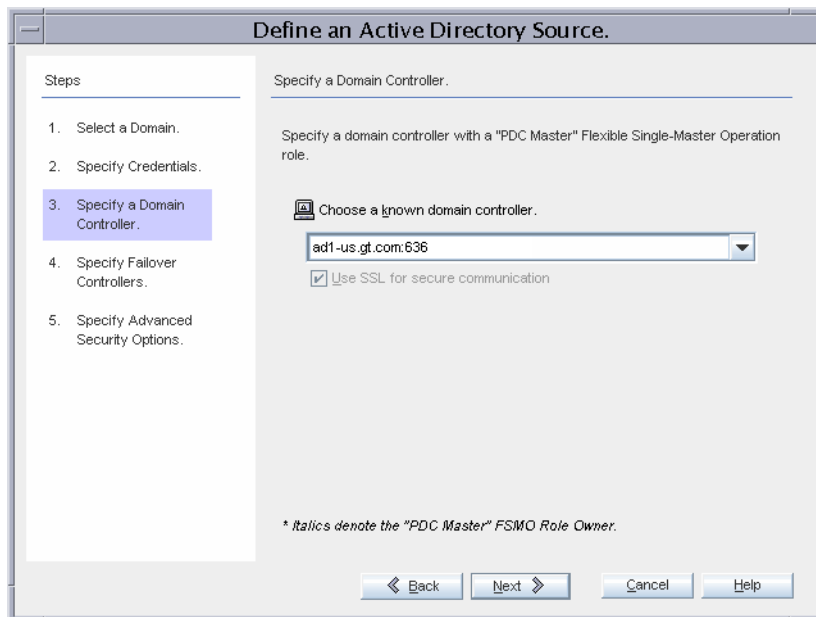


Because Global Telco requires the strictest security possible, the Directory Server Connector will require a trusted SSL certificate from the directory server, and the Identity Synchronization for Windows Directory Server Plug-ins will communicate over SSL to Active Directory.

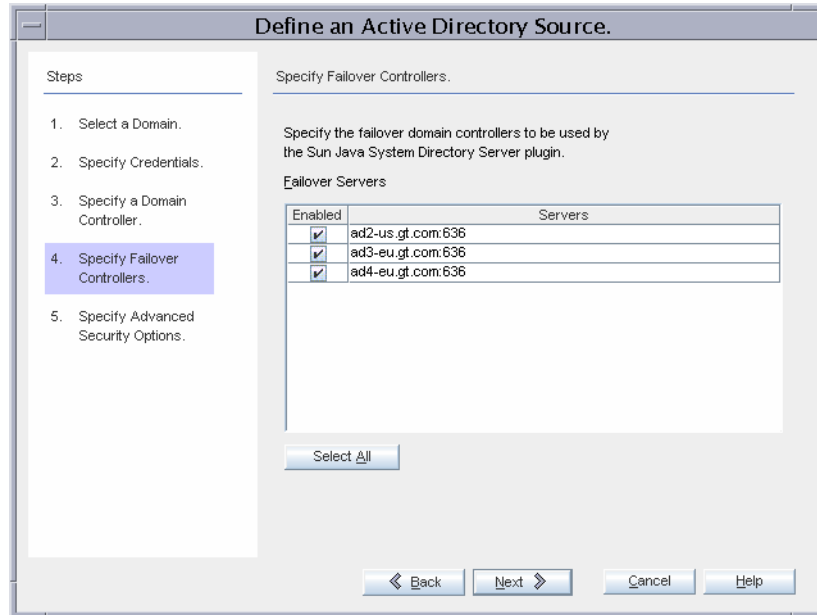
Note that the Identity Synchronization for Windows Plug-ins inherit the SSL configuration of the directory server. Therefore, if the Directory Server requires trusted certificates, the plug-in can only communicate with Active Directory if it provides a trusted certificate. Enabling these enhanced security options implies the following additional installation actions.



ad1-us.gt.com is the PDC FSMO role owner and is selected as the domain with which the controller for the Active Directory Connector will communicate. The connector communicates over SSL.

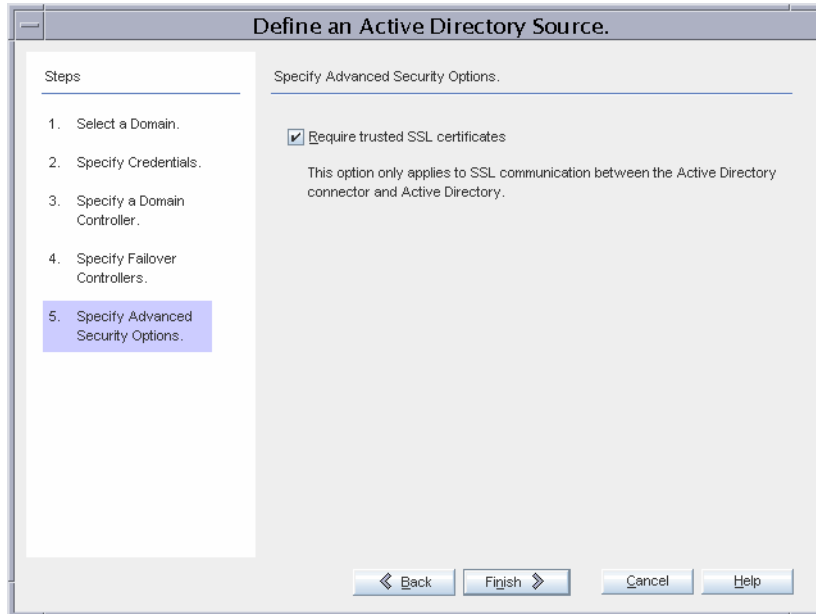


All three remaining domain controllers will be used for failover during on-demand password synchronization.

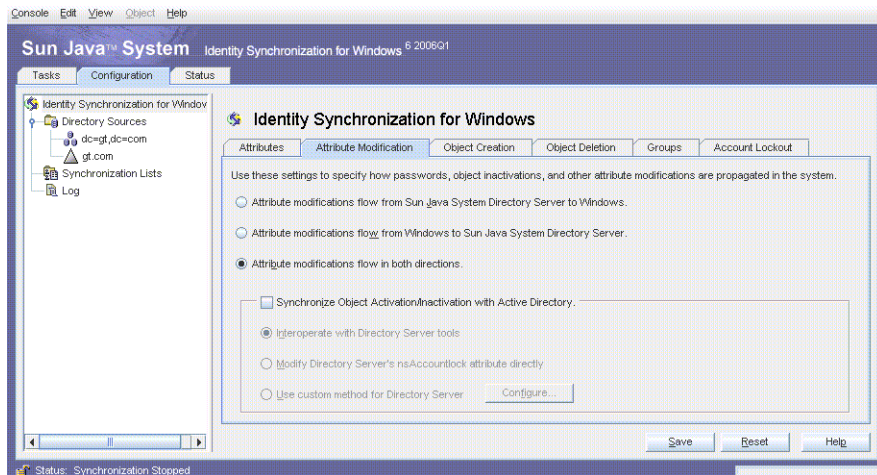


Because Global Telco requires the strictest security possible, the Active Directory Connector will require a trusted SSL certificate from ad1-us.gt.com.

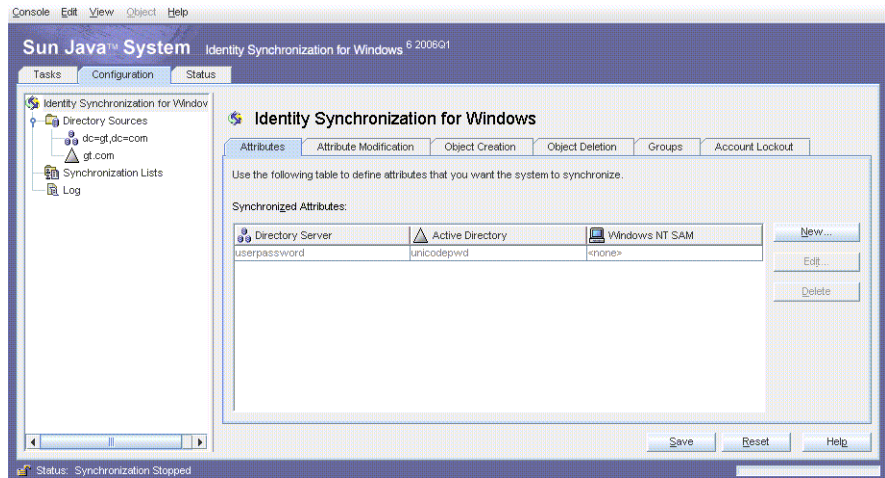
Enabling this advanced security option implies additional installation steps as outlined below.



The only default global setting that is changed is the synchronization of attribute modifications from Active Directory to Directory Server, and from Directory Server to Active Directory.

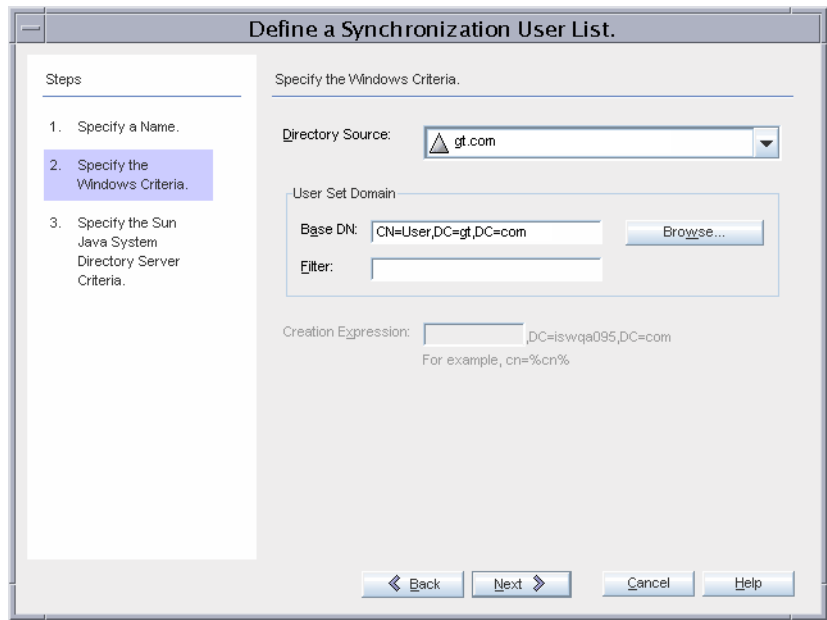


Only passwords are synchronized. No additional attributes are synchronized.

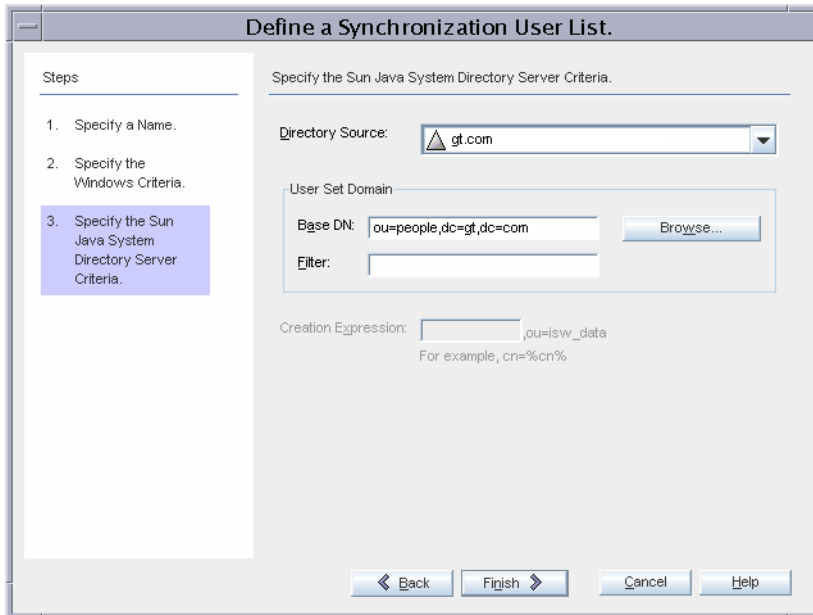


A single SUL, GT_USERS, is created as shown in “Primary Installation” on page 81.

Active Directory users are stored under the default `cn=users,dc=gt,dc=com` container. The existing users (Administrator, Guest, TsInternetUser, and iswUser) are excluded from synchronization.



The Directory Server users are stored in the default `ou=people,dc=gt,dc=com` container.



After the configuration is saved, each connector is installed on connectors-us.gt.com, and the Identity Synchronization for Windows Plug-in is installed.

```
bash-2.05# ./idsync printstat -w <password omitted> -q <password omitted>
Exploring status of connectors, please wait...
```

Connector ID: CNN100

```
Type: Sun Java(TM) System Directory
Manages: dc=gt,dc=com (ldaps://master1-us.gt.com:636)
(ldaps://master2-us.gt.com:636)
State: READY
Installed on: connectors-us.gt.com
Plugin SUBC100 is installed on ldaps://master1-us.gt.com:636
Plugin SUBC101 is installed on ldaps://master2-us.gt.com:636
Plugin SUBC102 is installed on ldaps://master3-eu.gt.com:636
Plugin SUBC103 is installed on ldaps://master4-eu.gt.com:636
Plugin SUBC104 is installed on ldaps://replica1-us.gt.com:636
Plugin SUBC105 is installed on ldaps://replica2-us.gt.com:636
Plugin SUBC106 is installed on ldaps://replica3-eu.gt.com:636
Plugin SUBC107 is installed on ldaps://replica4-eu.gt.com:636
```

Connector ID: CNN101

```
Type: Active Directory
Manages: gt.com (ldaps://ad2-us.gt.com:636) (ldaps://ad3-eu.gt.com:636)
(ldaps://ad4-eu.gt.com:636) (ldaps://ad1-us.gt.com:636)
State: READY
```

Installed on: connectors-us.gt.com

Sun Java(TM) System Message Queue Status: Started

Checking the System Manager status over the Sun Java(TM) System Message Queue.

System Manager Status: Started

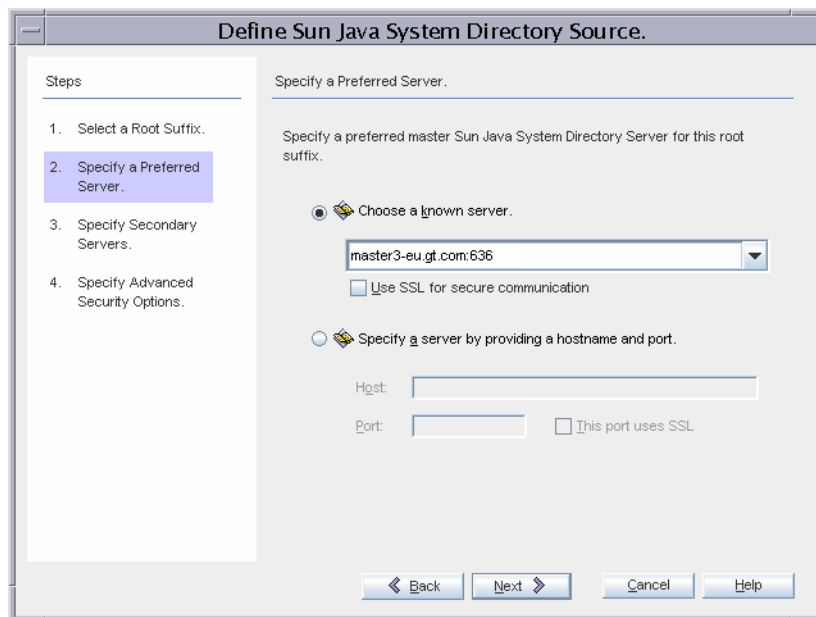
Remaining Installation and Configuration Steps:

1. Install the Sun Directory Server Plugin on every other master and read-only replica that manage users under dc=gt,dc=com.
 2. Run 'idsync resync' to establish links between existing Directory Server and Windows users.
 3. Start synchronization using the console or the 'idsync startsync' command.
- SUCCESS

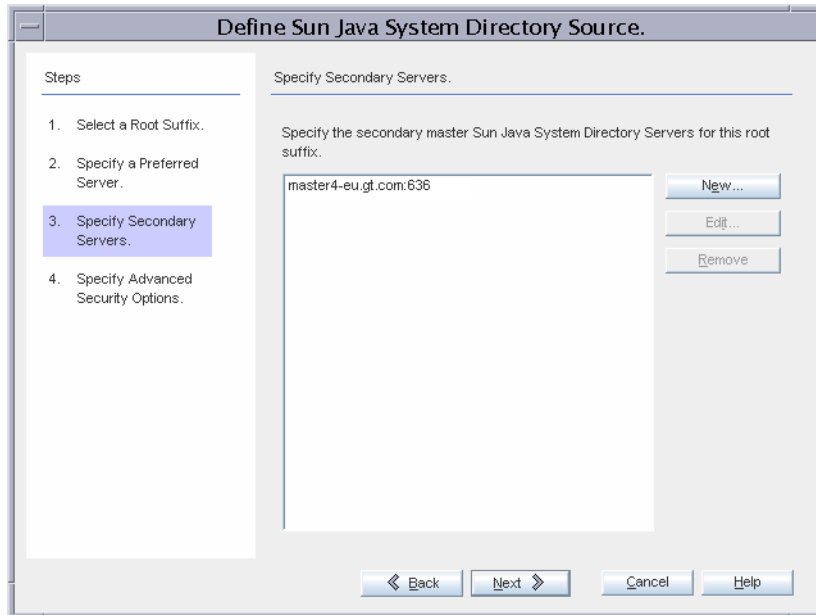
Failover Installation

After the primary installation is complete, you install the Identity Synchronization for Windows Core on config-eu.gt.com, and configure it using the console.

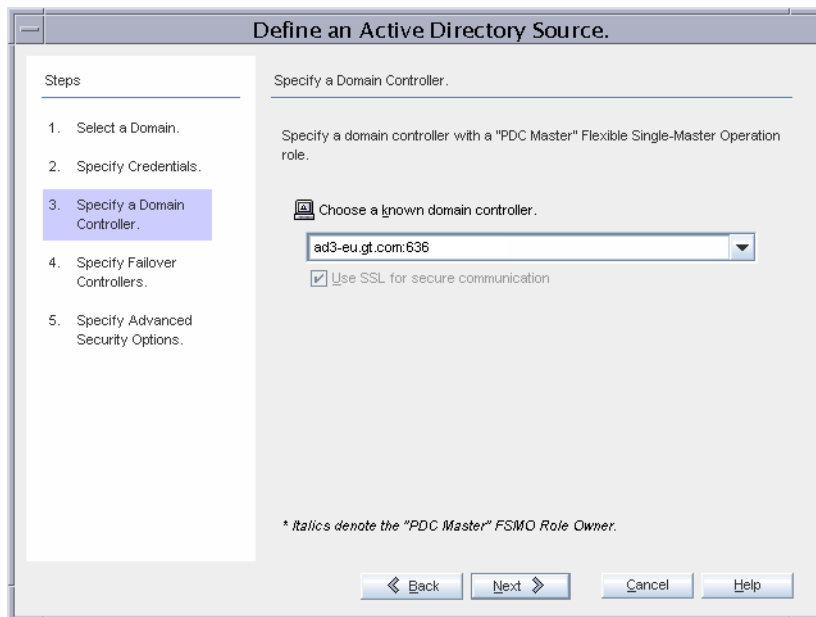
master3-eu.gt.com is the preferred Directory Server in the failover installation.



master4-eu.gt.com is the secondary Directory Server in the failover installation.

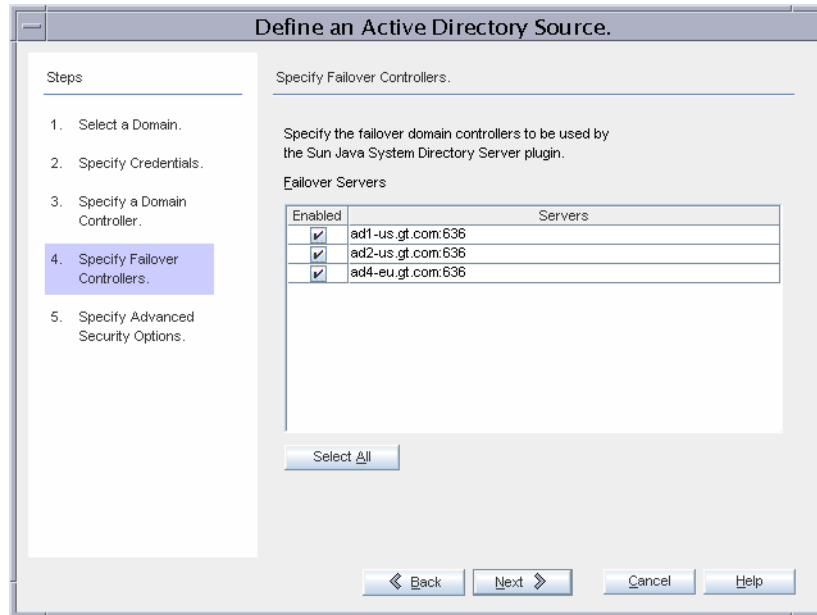


ad3-eu.gt.com is chosen as the domain controller with which the Active Directory Connector will communicate.



Note that a warning will be displayed stating that the password updates might become slow because `ad3-eu.gt.com` is not the PDC FSMO role owner. This warning can be ignored because changing the PDC FSMO role to this domain controller is part of the failover procedure. A similar warning is also displayed when the configuration is saved.

The remaining domain controllers are selected for failover during on-demand password synchronization.



```
bash-2.05# /opt/SUNWisdw/bin/idsync printstat -q <
omitted password\> -w <omitted password\>
Exploring status of connectors, please wait...
Connector ID: CNN100
Type: Sun Java(TM) System Directory
Manages: dc=gt,dc=com (ldaps://master3-eu.gt.com:636)
(ldaps://master4-eu.gt.com:636)
State: READY
Installed on: connectors-eu.gt.com

Connector ID: CNN101
Type: Active Directory
Manages: gt.com (ldaps://ad1-us.gt.com:636) (ldaps://ad2-us.gt.com:636)
(ldaps://ad4-eu.gt.com:636) (ldaps://ad3-eu.gt.com:636)
State: READY
Installed on: connectors-eu.gt.com
```

Sun Java(TM) System Message Queue Status: Started

Checking the System Manager status over the Sun Java(TM) System Message Queue.

System Manager Status: Started

Remaining Installation and Configuration Steps:

1. Install the Sun Directory Server Plugin at master ldaps://master3-eu.gt.com:636 by re-running the installer.
 2. Install the Sun Directory Server Plugin at master ldaps://master4-eu.gt.com:636 by re-running the installer.
 3. Install the Sun Directory Server Plugin on every other master and read-only replica that manage users under dc=gt,dc=com.
 4. Run 'idsync resync' to establish links between existing Directory Server and Windows users.
 5. Start synchronization using the console or the 'idsync startsync' command.
- SUCCESS

Setting Up SSL

Global Telco requires that all network traffic is encrypted, so SSL is used with trusted certificates for all LDAP connections. This setup includes connections between the following:

- Directory Server Connectors and the preferred and secondary Directory Servers
- Active Directory Connector and the Active Directory domain controller
- Preferred and secondary Directory Servers, and the Active Directory domain controllers for on-demand password synchronization
- All other preferred and secondary Directory Servers and read-only replicas, and the preferred and secondary Directory Servers for on-demand password synchronization.

Companies typically use only a few Certificate Authorities (CA) to sign certificates, so the simplest approach to setting up SSL for Identity Synchronization for Windows is to add all CA certificates to every component's certificate database. Global Telco uses two Certificate Authorities: one to sign its Directory Server certificates and one to sign its Active Directory certificates. Both of these CA certificates are added to the certificate database of each connector and each directory server instance.

Note – The `idsync certinfo` command displays the steps for configuring SSL for Identity Synchronization for Windows components, based on the current configuration. The command does not have access to each component's certificate database, so it cannot determine if the steps have already been followed.

The output of this command is shown for the following primary installation. The output for the failover installation is identical except that the roles of the U.S. and European systems are reversed.

```
bash-2.05# /opt/SUNWisw/bin/idsync certinfo -q <omitted password> -w <omitted password>  
Connector: CNN100
```

```
Installation Host: connectors-us
```

```
Installation Path: /opt
```

```
Certificate Database Location: /var/opt/SUNWisw/etc/CNN100
```

```
**The Directory Server Connector's certificate database must contain the  
CA certificate used to sign Directory Server's SSL certificate. If this  
certificate has not already been added to the connector's certificate  
database, please export the CA certificate and import into Directory Server  
Connector certificate database for server ldaps://master1-us.gt.com:636.
```

```
**The Directory Server's certificate database must contain the CA certificate  
used to sign the Active Directory's SSL certificate. If this certificate has  
not already been added to the Directory Server's certificate database, please  
export the CA certificate from the Active Directory at ldaps://ad1-us.gt.com:636  
and import into Directory Server certificate database for server  
ldaps://master1-us.gt.com:636.
```

```
**The Directory Server's certificate database must contain the CA certificate  
used to sign the Active Directory's SSL certificate. If this certificate has  
not already been added to the Directory Server's certificate database, please  
export the CA certificate from the Active Directory at ldaps://ad2-us.gt.com:636  
and import into Directory Server certificate database for server  
ldaps://master1-us.gt.com:636.
```

```
**The Directory Server's certificate database must contain the CA certificate  
used to sign the Active Directory's SSL certificate. If this certificate has  
not already been added to the Directory Server's certificate database, please  
export the CA certificate from the Active Directory at ldaps://ad3-eu.gt.com:636  
and import into Directory Server certificate database for server  
ldaps://master1-us.gt.com:636.
```

```
**The Directory Server Connector's certificate database must contain the  
CA certificate used to sign the Directory Server's SSL certificate.  
If this certificate has not already been added to the connector's certificate  
database, please export the CA certificate and import into Directory Server  
Connector certificate database for server ldaps://master2-us.gt.com:636.
```

```
**The Directory Server's certificate database must contain the  
CA certificate used to sign the Active Directory's SSL certificate.
```

If this certificate has not already been added to the Directory Server's certificate database, please export the CA certificate from the Active Directory at `ldaps://ad1-us.gt.com:636` and import into Directory Server certificate database for server `ldaps://master2-us.gt.com:636`.

****The Directory Server's certificate database must contain the CA certificate used to sign the Active Directory's SSL certificate. If this certificate has not already been added to the Directory Server's certificate database, please export the CA certificate from the Active Directory at `ldaps://ad2-us.gt.com:636` and import into Directory Server certificate database for server `ldaps://master2-us.gt.com:636`.**

****The Directory Server's certificate database must contain the CA certificate used to sign the Active Directory's SSL certificate. If this certificate has not already been added to the Directory Server's certificate database, please export the CA certificate from the Active Directory at `ldaps://ad4-eu.gt.com:636` and import into Directory Server certificate database for server `ldaps://master1-us.gt.com:636`.**

****The Directory Server's certificate database must contain the CA certificate used to sign the Active Directory's SSL certificate. If this certificate has not already been added to the Directory Server's certificate database, please export the CA certificate from the Active Directory at `ldaps://ad3-eu.gt.com:636` and import into Directory Server certificate database for server `ldaps://master2-us.gt.com:636`.**

****The Directory Server's certificate database must contain the CA certificate used to sign the Active Directory's SSL certificate. If this certificate has not already been added to the Directory Server's certificate database, please export the CA certificate from the Active Directory at `ldaps://ad4-eu.gt.com:636` and import into Directory Server certificate database for server `ldaps://master2-us.gt.com:636`.**

Connector: CNN101
Installation Host: connectors-us
Installation Path: /opt
Certificate Database Location: /var/opt/SUNWsw/etc/CNN101

****The Active Directory Connector's certificate database must contain the CA certificate used to sign the Active Directory's SSL certificate. If this certificate has not already been added to the Active Directory Connector certificate database,**

please export the CA certificate from the Active Directory and import into Active Directory Connector's certificate database for server
ldaps://ad1-us.gt.com:636.

****The Active Directory Connector's certificate database must contain the CA certificate used to sign the Active Directory's SSL certificate. If this certificate has not already been added to the Active Directory Connector certificate database, please export the CA certificate from the Active Directory and import into Active Directory Connector's certificate database for server
ldaps://ad2-us.gt.com:636.**

****The Active Directory Connector's certificate database must contain the CA certificate used to sign the Active Directory's SSL certificate. If this certificate has not already been added to the Active Directory Connector certificate database, please export the CA certificate from the Active Directory and import into Active Directory Connector's certificate database for server
ldaps://ad3-eu.gt.com:636.**

****The Active Directory Connector's certificate database must contain the CA certificate used to sign the Active Directory's SSL certificate. If this certificate has not already been added to the Active Directory Connector certificate database, please export the CA certificate from the Active Directory and import into Active Directory Connector's certificate database for server
ldaps://ad4-eu.gt.com:636.**

SUCCESS

“[Setting Up SSL](#)” on page 91 summarizes SSL communication between components in this installation, including trust requirements for the primary and failover installations.

The following table provides more detailed information.

TABLE 3-1 SSL Communication Between Components

Component	Must Trust Certificates From	Required By	Comments
Directory Server Connector on connector-us.gt.com	master1-us.gt.com	Primary	Only required if the Require Trusted SSL Certificates check box is selected in the Console.
	master2-us.gt.com		

TABLE 3-1 SSL Communication Between Components (Continued)

Component	Must Trust Certificates From	Required By	Comments
Active Directory Connector on connector-us.gt.com	ad1-us.gt.com	Primary	Only required if the Require Trusted SSL Certificates check box is selected in the Console. Note – The output of <code>idsync certinfo</code> erroneously mentions that certificates for the other Active Directory domain controllers are required.
Directory Server Connector on connector-eu.gt.com	master3-eu.gt.com	Failover	Only required if the Require Trusted SSL Certificates check box is selected in the Console.
	master4-eu.gt.com		
Active Directory Connector on connector-eu.gt.com	ad3-eu.gt.com	Primary	Only required if the Require Trusted SSL Certificates check box is selected in the Console. Note – The output of <code>idsync certinfo</code> erroneously mentions that certificates for the other Active Directory domain controllers are required.
master1-us.gt.com	ad1-us.gt.com	Primary	Required for on-demand password synchronization.
	ad2-us.gt.com		
	ad3-us.gt.com		
	ad4-us.gt.com		
	master3-eu.gt.com	Failover	Required for on-demand password synchronization. The <code>Idsync certinfo</code> command does not mention this requirement.
	master4-eu.gt.com		
master2-us.gt.com	ad1-us.gt.com	Primary	Required for on-demand password synchronization.
	ad2-us.gt.com		
	ad3-us.gt.com		
	ad4-us.gt.com		
	master3-eu.gt.com	Failover	Required for on-demand password synchronization. The <code>idsync certinfo</code> does not mention this requirement.
	master4-eu.gt.com		

TABLE 3-1 SSL Communication Between Components *(Continued)*

Component	Must Trust Certificates From	Required By	Comments
master3-eu.gt.com	ad1-us.gt.com	Failover	Required for on-demand password synchronization.
	ad2-us.gt.com		
	ad3-us.gt.com		
	ad4-us.gt.com		
	master1-us.gt.com	Primary	Required for on-demand password synchronization. The <code>idsync certinfo</code> command does not mention this requirement.
	master2-us.gt.com		
master4-eu.gt.com	ad1-us.gt.com	Failover	Required for on-demand password synchronization.
	ad2-us.gt.com		
	ad3-us.gt.com		
	ad4-us.gt.com		
	master1-us.gt.com	Primary	Required for on-demand password synchronization. The <code>idsync certinfo</code> command does not mention this requirement.
	master2-us.gt.com		
replica1-us.gt.com	master1-us.gt.com	Primary	Required for on-demand password synchronization. The <code>idsync certinfo</code> command does not mention this requirement.
replica2-us.gt.com	master2-us.gt.com		
replica3-eu.gt.com	master3-eu.gt.com	Failover	Required for on-demand password synchronization. The <code>idsync certinfo</code> command does not mention this requirement.
replica4-eu.gt.com	master4-eu.gt.com		

In this installation, Global Telco adds both of the CA certificates to the certificate databases of the four connectors and the eight directory servers.

Note – See the *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide* for detailed instructions on adding certificates to the certificate databases. The Directory Server and connectors must be restarted after the certificates have been added. The Directory Server must be restarted after the Identity Synchronization for Windows Plug-in is installed. Therefore, add the CA certificates to the Directory Servers' certificate databases *before* the Identity Synchronization for Windows Plug-in is installed.

Increasing Connector Worker Threads

By default each Directory Server and Active Directory connector uses four worker threads to apply changes. This number is increased in the Global Telco deployment to improve connector performance, especially during `idsync resync` operations. The number of connector threads is stored in the configuration directory in the `pswNumOutboundConnectorThreads` attribute, which is in the `pswSunDirectoryGlobals` entry and the `pswActiveDirectoryGlobals` entry. Before manually editing the configuration, all Identity Synchronization for Windows consoles must be closed.

- To find the `pswSunDirectoryGlobals` entry, the following command is used:

```
bash-2.05# ./ldapsearch -b "ou=identitysynchronization,ou=services,dc=gt,dc=com" -D "cn=Directory Manager" -w <omitted password>
"(&(objectclass=pswSunDirectoryGlobals)(pswversion\>=0))"
pswNumOutboundConnectorThreads
```

```
dn: cn=136,ou=Sun,ou=Globals,cn=active[13],ou=GlobalConfig,ou=1.1,ou=IdentitySynchronization,ou=Services,dc=gt,dc=com
pswNumOutboundConnectorThreads: 4
```

The entry that must be modified is shown here:

```
cn=136,ou=Sun,ou=Globals,cn=active[13],ou=GlobalConfig,ou=1.1,ou=IdentitySynchronization,ou=Services,dc=gt,dc=com.
```

- To find the `pswActiveDirectoryGlobals` entry, the following command is used:

```
bash-2.05# ./ldapsearch -b "ou=identitysynchronization,ou=services,dc=gt,dc=com" -D "cn=Directory Manager" -w <omitted password>
"(&(objectclass=pswActiveDirectoryGlobals)(pswversion\>=0))"
pswNumOutboundConnectorThreads
```

```
dn: cn=110,ou=ActiveDirectory,ou=Globals,cn=active[13],ou=GlobalConfig,ou=1.1,ou=IdentitySynchronization,ou=Services,dc=gt,dc=com
pswNumOutboundConnectorThreads: 4
```

The entry that must be modified is shown here:

```
cn=110,ou=ActiveDirectory,ou=Globals,cn=active[13],ou=GlobalConfig,ou=1.1,ou=IdentitySynchronization,ou=Services,dc=gt,dc=com.
```

These two entries are modified to increase the number of threads to a maximum of 20:

```
bash-2.05# ./ldapmodify -D "cn=Directory Manager" -w <omitted password>
```

```
dn: cn=136,ou=Sun,ou=Globals,cn=active[13],ou=GlobalConfig,ou=1.1,ou=IdentitySynchronization,ou=Services,dc=gt,dc=com
```

```
changetype: modify
replace: pswNumOutboundConnectorThreads
pswNumOutboundConnectorThreads: 20

modifying entry cn=136,ou=Sun,ou=Globals,cn=active[13],ou=GlobalConfig, ou=1.1,
ou=IdentitySynchronization,ou=Services,dc=gt,dc=com
dn: cn=110,ou=ActiveDirectory,ou=Globals,cn=active[13],ou=GlobalConfig, ou=1.1,
ou=IdentitySynchronization,ou=Services,dc=gt,dc=com
changetype: modify
replace: pswNumOutboundConnectorThreads
pswNumOutboundConnectorThreads: 20
modifying entry cn=110,ou=ActiveDirectory,ou=Globals,cn=active[13],
ou=GlobalConfig,ou=1.1,ou=IdentitySynchronization,ou=Services,dc=gt,dc=com
```

After these values are changed, the Identity Synchronization for Windows daemon on the system with the Identity Synchronization for Windows Core, is restarted to notify the System Manager to pick up the new configuration.

Note – Increasing the number of connector threads also increases the maximum number of LDAP connections that the connector will keep open to the directory.

Aligning Primary and Failover Configurations

Even though the primary and failover installations have similar configurations, some generated configuration parameters differ between the two deployments:

- The password used by the Directory Server Connector's `uid=PSWConnector,dc=gt,dc=com` user

Although this user is created when Directory Server is “prepared” for synchronization, the password is set when the Directory Server Connector is installed. The randomly generated password is set in the directory server entry and then stored in the configuration.

- The encryption key used by the Identity Synchronization for Windows Plug-in to encrypt passwords in the retro changelog

This key is randomly generated when the configuration is first saved.

Both of these values are encrypted and stored in the configuration directory with the rest of the Identity Synchronization for Windows configuration. However, the values cannot be copied between the two configurations because the encrypted values are unique to each deployment.

The limitation for the `uid=PSWConnector` entry has a workaround because Directory Server allows an entry to have multiple password values. During the installation process, the `uid=PSWConnector` entry can be manually modified to store the password used in the primary configuration and the password used in the failover configuration.

However, because the same encryption key cannot be used for both configurations, some password changes might be lost during failover. The failover process includes reinstalling the Identity Synchronization for Windows Plug-ins on each directory server so that they receive their configuration from the failover installation instead of the primary installation. Any password change made in Directory Server during this period will be lost. Identity Synchronization for Windows will log a message about the lost password.

Setting Multiple Passwords for uid=PSWConnector

After installing the Directory Server Connector for the primary installation, but before installing the Directory Server Connector for the failover installation, the password for the uid=PSWConnector user must be retrieved and saved:

```
bash-2.05# ./ldapsearch -h master1-us -b "dc=gt,dc=com" -D "cn=Directory Manager"
-w <omitted password> " (uid=PSWconnector)" userpassword
version: 1
dn: uid=PSWConnector,dc=gt,dc=com
userpassword: {SSHA}OUYr10Y2mHIyZfyVLM400nYi4UZGNSAVLAERRg==
```

{SSHA}OUYr10Y2mHIyZfyVLM400nYi4UZGNSAVLAERRg== is the password that the primary Directory Server Connector uses to connect to the directory server. Installing the Directory Server Connector for the failover installation overwrites this password. At this point, retrieve the entry again:

```
bash-2.05# ./ldapsearch -h master1-us -b "dc=gt,dc=com" -D "cn=Directory Manager"
-w <omitted password> " (uid=PSWconnector)" userpassword
version: 1
dn: uid=PSWConnector,dc=gt,dc=com
userpassword: {SSHA}k9AFSUGsY1NK038PvIB4LJzVNB0sQHh4JHJXFQ==
```

{SSHA}k9AFSUGsY1NK038PvIB4LJzVNB0sQHh4JHJXFQ== is the password that the failover Directory Server Connector uses to connect to the Directory Server. At this point, the Directory Server Connector for the primary installation can no longer log in to the directory, so modify the entry to include both passwords.

```
bash-2.05# ./ldapmodify -h master1-us -D "cn=Directory Manager"
-w <omitted password>
dn: uid=PSWConnector,dc=gt,dc=com
changetype: modify
replace: userpassword
userpassword: {SSHA}OUYr10Y2mHIyZfyVLM400nYi4UZGNSAVLAERRg==
userpassword: {SSHA}k9AFSUGsY1NK038PvIB4LJzVNB0sQHh4JHJXFQ==
modifying entry uid=PSWConnector,dc=gt,dc=com
```

After this process is complete, both Directory Server Connectors will be able to log in to the directory. To verify this, stop and restart the Identity Synchronization for Windows daemon for

the primary installation on `connectors-us.gt.com`, and for the failover installation on `connectors-us.gt.com`. After the connectors start and receive their configuration, they will open a connection to the directory. If there are any problems with the credentials, they are reported in the central logs.

Note – Every time the Directory Server Connector is installed, a new password is generated and written to the `uid=PSWConnector` entry. If Directory Server Connector is uninstalled and reinstalled, this procedure must be followed again. Also, if the Directory Server Connector for the failover installation was installed before the primary `uid=PSWConnector` password was retrieved, save the current `uid=PSWConnector` password (for the failover configuration), uninstall and reinstall the primary Directory Server Connector, and then retrieve the current `uid=PSWConnector` password (for the primary configuration).

Initial idsync resync Operations

The next step in the installation and configuration process is to run `idsync resync`. In this deployment, `idsync resync` takes these actions:

- Establishes links between existing Active Directory and Directory Server users
- Resets the Directory Server password to the Active Directory password by forcing on-demand password synchronization the first time that a user logs into Directory Server.
- Primes the Active Directory Connector's object cache so that changes to existing users are not falsely interpreted as newly created users

Performing a full `idsync resync` operation is expensive in Identity Synchronization for Windows. For a deployment as large as `gt.com` (500,000 users), the operation might take a few hours to complete. It also places a heavy load on Message Queue because all users and log messages are sent over Message Queue during the `idsync resync` operation. To reduce this load, do the following:

1. Increase the Message Queue Broker available memory to 1 Gbyte as described in the *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide*.
2. Change the Identity Synchronization for Windows system-wide log level from the default FINE to INFO, using the Console.

Note – If you need to increase the log level to debug an issue, use the `-a` option of `idsync resync` to restrict the operation to only the affected users.

3. Divide the 500,000 users into five separate `idsync resync` batches by using the `-a` option. This process is described in more detail in the next section.

These steps are also performed for the failover installation.

Initial idsync resync Operation for Primary Installation

The first time that you run `idsync resync`, divide the Active Directory users into five batches to reduce the peak load on Message Queue. Use the `-a <ldap-filter>` option with `idsync resync` to limit the scope of the `idsync resync` operation to a subset of Active Directory users. When dividing users, the following must be ensured:

- The union of the LDAP filters includes every user that should be synchronized.
For example, all characters might not be English or even ASCII, so resynchronization based on the 26 letters of the English alphabet will skip users with non-English characters.
- The LDAP filters can overlap but a significant overlap will increase the total running time of the operation.
- Batch sizes are not too large, to keep the Message Queue Broker reaching its memory limits.
- The attributes in the filter are indexed in Active Directory.
Consult Microsoft's documentation to determine which Active Directory attributes are indexed. The standard set of indexed attributes includes `cn`, `sn`, and `samaccountname`.
- Certain types of searches on indexed attributes can be expensive.
For example, an absence search, `(! (sn=*))`, will result in a time-out error even though it is indexed.
- If the filter includes attributes that are not mandatory, for example, `sn`, all entries must have a value for that attribute so that entries are not overlooked.
- If a user attribute used in the filter can change value during the operation, users whose attributes change might not be synchronized.
For example, if an entry's `sn` value changes from Smith to Anderson after the first batch of entries has been synchronized, this user will be skipped. The following script exploits Active Directory's `uSNchanged` attribute to resynchronize all entries that changed during the sequence of `idsync resync` operations.

Note – Running `idsync resync` in batches will not significantly increase the total time required to do a full `idsync resync` operation. It might even speed the operation because it prevents the Message Queue Broker from thrashing as it reaches its memory limits.

Only a single `idsync resync` operation can be run at a time, not in parallel.

The `-a <ldap-filter>` option always applies to the source of the `idsync resync` operation. When users are linked with the `-f` option, Active Directory must be the source of the operation. Thus, only consider how to divide users based on the available Active Directory attributes.

Due to these requirements, Global Telco is resynchronizing its users based on each user's `cn` attribute, a mandatory attribute that is indexed. Users are categorized into the following groups:

- `cn <= a`

- a <= cn <= d
- d <= cn <= h
- h <= cn <= m
- m <= cn <= s
- s <= cn

Note – If a user's cn exactly matches one of the boundary conditions (for example, cn=d), the user will be synchronized twice, introducing negligible overhead but not causing any problems.

Global Telco then uses the following Perl script to iterate through the idsync resync operations. The resync-batch.pl script first retrieves the current maximum uSNChanged value from Active Directory. The specified Active Directory host must be the same host as the one the Active Directory Connector uses for communication because uSNChanged values differ between Active Directory domain controllers. The script then iterates through each idsync resync operation, pausing for one minute between the operations to allow the connectors to reset and settle. The last search filter matches all user entries that changed during the other idsync resync operations.

```
#!/usr/bin/perl
#
# This script executes a sequence of resync operations.
# Each resync operation occurs on a subset of users
# based on an LDAP filter.
#
# Parameters specific to this deployment
#
my $linkFile = "/var/opt/SUNWsw/samples/linkusers-simple-gt.cfg";
my $idsync = "/opt/SUNWsw/bin/idsync";
my $adHost = "ad1-us.gt.com";
my $secBetweenOps = 60;
my $dsPassword = "<omitted password>";
my $configPassword = "<omitted password>";

# Guard against users whose cn changes during this operation
# by retrieving the maximum uSNChanged value before we start
# the operation, and later running resync for only these users.
my $origUsn = "ldapsearch -h $adHost -b '' -s base '(objectclass=*)' \\
highestCommittedUSN | grep highestCommittedUSN | sed 's/^0-9//g'";
chomp $origUsn;
# Run each user in a batch
my @BATCH_FILTERS = (
    (cn<=a)",          # cn <= a
    "(&(cn\>=a)(cn<=d))", # a <= cn <= d
    "(&(cn\>=d)(cn<=h))", # d <= cn <= h
    "(&(cn\>=h)(cn<=m))", # h <= cn <= m
```

```

        "(&(cn\>=m)(cn<=s))", # m <= cn <= s
        "
        (cn\>=s)",          # s <= cn
        "(uSNChanged\>=$origUsn)", # modified users);
# Run each user in a batch
for my $filter (@BATCH_FILTERS) {
    print "Running resync with filter '$filter'.\n";
    my $command = "$idsync resync -a '$filter' -f $linkFile."
        "-i NEW_LINKED_USERS -q $configPassword -w $dsPassword";
    system($command);
    print "Waiting $secBetweenOps before next iteration.\n";
    sleep $secBetweenOps;
}

```

Notice that the `-i NEW_LINKED_USERS` option is passed to `idsync resync`, which resets all Directory Server passwords to match their Active Directory counterparts after the entries are linked.

Note – If Global Telco did not want to reset the passwords for the Directory Server accounts, it would *not* run this command with the `-k` option, which only links users. The `idsync resync` command primes the object cache database in the Active Directory Connector, which is used to detect changes to entries.

When `idsync resync` is run with the `-k` option, the object cache is not primed, and if the connector detects a change to an existing user, it will erroneously assume that the change is a new user. Thus, if you run `idsync resync -k`, you must run the `idsync resync` command again with the `-u` option, which updates the object cache only.

Initial idsync resync Operation for Failover Installation

You must also run the `idsync resync` command for the failover installation. The links between the entries are already established, and `idsync resync` is only needed to prime the object cache database for the Active Directory Connector. Therefore, the command is run with the `-u` option.

To avoid overloading Message Queue with log messages, the following must be done:

- Increase the Message Queue Broker maximum memory
- Change the Identity Synchronization for Windows log level to INFO
- Run `idsync resync` over the entire user population in batches

You must modify the script in the previous section and run it on the failover installation Core, `config-eu.gt.com`. Change the parameters at the top of the script, and replace the `-i NEW_LINKED_USERS` option with just `-u`.

Periodic idsync resync Operations

Periodic idsync resync Operation for Primary Installation

As Global Telco uses Identity Manager to provision users and has not configured Identity Synchronization for Windows to synchronize the creation of new users, idsync resync must be run periodically to establish links between recently created users. If these users are not linked, modifications to their passwords will not synchronize. Running a full idsync resync operation for 500,000 users could take a few hours because synchronization of new changes is delayed during an idsync resync operation. It is unacceptable to have Identity Synchronization for Windows offline for so long, so Global Telco uses the Active Directory usnCreated attribute to synchronize only those users that have been created recently. The administrator uses the following resync-recent.pl script, which is run every hour, to synchronize users that have been created in the last three days. This script is run on users created in the last three days instead of only the last hour because a new employee might not have a Directory Server account until a few days after the Active Directory account is created.

The script is run periodically with the following arguments, using cron:

```
resync-recent.pl ad1-us.gt.com usnCreated
-f /var/opt/SUNWiw/samples/linkusers-simple-gt.cfg
-i NEW_LINKED_USERS -b -w - -q - < /var/opt/SUNWiw/passwords
```

The first argument is the name of the Active Directory domain controller that the connector communicates with. The second argument determines whether the idsync resync command should be run only on entries that were created recently. The remaining arguments are passed directly to resync. In this case, the Directory Server password is reset for all users that are newly linked. The - value used for the two password options directs the idsync resync command to read the values from STDIN; which prevents passwords from appearing in the command line and being available to anyone on the system using commands such as ps. The configuration directory password and the configuration password are written to /var/opt/SUNWiw/passwords, and this file is then protected with the strictest file system permissions.

resync-recent.pl script:

```
#!/usr/bin/perl

# This sample script shows how to run idsync resync only on users
# that have changed recently and can be used to reduce the
# impact of running resync frequently. By default it
# only resync's users that were modified or created in
# the last three days. The script stores a daily history
```



```

# of Active Directory's highestCommittedUSN attribute.
#
# The arguments of the command are
#
# <Active Directory-domain-controller-name>
# (usnChanged|usnCreated)
# [args-for-resync]
#
# To link users that were created recently run the command
# ad.example.com usnCreated -k -f link.cfg -q <pw> -q <pw>
#
# To synchronize all users that were modified recently
# ad.example.com usnChanged -q <pw> -q <pw>
#
# To prime the object cache for users that were modified recently
# ad.example.com usnChanged -u -q <pw> -q <pw>
#
# NOTE: this script is only provided as a guide.
# It must be adapted to the specific Identity
# Synchronization for Windows environment by
# changing the paths and options below as appropriate
# and adding additional error handling code.
#
my $USAGE = "USAGE: resync-recent.pl ".
    "<Active Directory-domain-controller-name> ".
    "(usnChanged|usnCreated) [args-for-resync]\\n";

my $IDSYN = "/opt/SUNWisw/bin/idsync";
my $USN_FILE = "/opt/SUNWisw/bin/usnHistory";
my $MAX_DAYS_HISTORY = 3;

#
# SCRIPT BEGIN
#

#
# Argument parsing
#
my $adDomainController = shift @ARGV or die "$USAGE";
my $usnSearchAttr = shift @ARGV or die "$USAGE";
$usnSearchAttr =~ /usnChanged/i or
    $usnSearchAttr =~ /usnCreated/i or
    die "$USAGE\\n";
my $resyncArgs = getArgsAsString(@ARGV);

#

```

```
# Read the highestCommittedUSN history and add today's
# date to it if not it's there and then write the history
# out.
#
my %usnHistory = readUsnHistory();
my $today = getCurrentDate();
if (!$usnHistory{$today}) {
    $usnHistory{$today} = getHighestUsn($adDomainController);
}
writeUsnHistory(%usnHistory);

#
# Run the resync command based on the oldest usnChanged
# value in the history.
#
my $oldestUsn = getOldestUsn(%usnHistory);
my $filter = "($usnSearchAttr\>=$oldestUsn)";
my $resyncCmd = "IDSYNC resync -a \"$filter\" $resyncArgs";
print "Running $resyncCmd\n";
system($resyncCmd);

#
# SCRIPT END
#

#
# Return the current date as a string, e.g. 2004/03/04
#
sub getCurrentDate {
    my ($day, $month, $year) = (localtime(time))[3,4,5];
    $month++;
    $year += 1900;
    return sprintf "%d/%02d/%02d", $year, $month, $day;
}
#
# Searches the root DSE at the specified host and returns
# the highestCommittedUSN value.
#
sub getHighestUsn {
    my $adHost = shift @_;
    my $cmd = "ldapsearch -h $adHost -b \"\" -s base ".
        "\"\"(objectclass=*)\" highestCommittedUSN | ".
        "grep highestCommittedUSN | sed \"s/[^0-9]//g\"";
    my $highestUsn = `$cmd`;
    chomp $highestUsn;
    print "highestCommittedUSN at $adHost is $highestUsn.\n";
    return $highestUsn;
}
```

```

#
# Converts the command line args into a string.
#
sub getArgsAsString {
    my $args = "";
    for my $arg (@_) {
        $args .= " \"$arg\"";
    }
    return $args;
}

#
# Returns the oldest usnChanged value from the history.
#
sub getOldestUsn {
    my %usnHistory = @_;

    my @dates = getHistoryDates(%usnHistory);

    # Return the last element.
    return $usnHistory{$dates[$#dates]};
}

#
# Return a sorted list of the dates in the history.
#
sub getHistoryDates {
    my %history = @_;
    return reverse sort(keys %usnHistory);
}

#
# Writes the most recent daily history of highestCommittedUSN.
# No more than $MAX_DAYS_HISTORY days history is recorded.
#
sub writeUsnHistory {
    my %usnHistory = @_;
    if (!open(OUT, ">$USN_FILE")) {
        print STDERR "Could not open $USN_FILE for writing.\n";
        return;
    }

    # Write the history up to the last $MAX_DAYS_HISTORY days.
    my @dates = getHistoryDates(%usnHistory);
    for (my $i = 0;
        $i <= $#dates and $i < $MAX_DAYS_HISTORY;
        $i++)

```

```
        {
            my $date = $dates[$i];
            print OUT "$date:$usnHistory{$date}\\n";
        }

        close(OUT);
    }

#
# Reads the daily history of highestCommittedUSN
#

sub readUsnHistory {
    my %usnHistory = ();
    if (!open(IN, "<$USN_FILE")) {
        print STDERR "Could not read history from $USN_FILE.\\n";
        return %usnHistory;
    }

    while (my $line = <IN>) {
        chomp $line;
        my ($date, $usn) = split /:/, $line;
        $usnHistory{$date} = $usn;
    }
    close(IN);

    return %usnHistory
}
```

An alternative to running `idsync resync` periodically is to modify the process to set the necessary link information when the Directory Server entry is processed. The process is straightforward:

1. Add the `dpswuser` object class to the user entry.
2. Set the `dpswuserlink` attribute to match the user's `objectguid` attribute (from Active Directory).
3. (Optional) Set the `dpswvalidate` attribute to `true` to force on-demand password synchronization.

Periodic idsync resync Operation for Failover Installation

To speed the failover process, the `idsync resync` operation is run periodically on the failover installation to keep the object cache database in the Active Directory Connector up-to-date. If the object cache is not kept up-to-date, the Active Directory Connector will detect and

propagate many changes that were already synchronized by the primary installation. Not keeping the object cache database up-to-date will also significantly increase the load, and place a heavier load on Directory Server during the failover scenario.

The same `resync-recent.pl` script that is used in the primary installation is used in this installation except that it is run from the system that contains the failover installation Core, `config-eu.gt.com`. The `cron` command is used to run the script daily with the following arguments. The `-u` option is specified to update only the Active Directory Connector's object cache.

```
resync-recent.pl ad3-eu.gt.com usnCreated -u -b -w - -q - < /var/opt/SUNWisdw/passwords
```

Note – The more often this script is run in the failover environment, the more likely changes will be lost during the failover process. `idsync resync -u` should not be run after the primary installation fails. If the command is run often (for example, every hour), it is likely that it will be run while the primary installation has failed, but the failure has not yet been detected. As this script keeps track of a three-day history of the `highestCommittedUSN` values, it could be updated to search for entries that were modified in the last three days but not modified in the last day. As long as the primary installation failure is detected within one day and the `cron` job of this script was stopped, no Active Directory changes are lost.

Configuring Identity Manager

For details on configuring the Sun Java System Identity Manager to coexist with Identity Synchronization for Windows, see [Appendix B](#).

Understanding the Failover Process

The goal of the failover process is to avoid losing any changes that occurred between the time the primary system went down and the failover system was brought up. If changes are lost, Identity Synchronization for Windows should at least report the changes that are lost.

After synchronization is started for the first time, an Identity Synchronization for Windows Connector can continue to detect changes that occur in these situations:

- When synchronization is stopped
- When the connector process is not running, for example, during a system restart
- When the network connection between the connector and the directory source is down
- During a `idsync resync` operation

The Directory Server and Active Directory connectors use similar mechanisms to achieve this goal.

Directory Server Connector

The Directory Server Connector persistently stores the changenumber of the last retro changelog entry that it has processed. When the connector begins detecting changes again, it processes all changes that occurred when it was not running, starting with the last changenumber that it processed. This changenumber is stored in the connector's persist directory, for example, `/var/opt/SUNWsw/persist/ADP100/accessor.state`. If this file does not exist, the connector detects only new changes in Directory Server.

This way of processing the changes has an effect on the failover process. If synchronization is never started for the failover installation, the file will not exist, and the Directory Server Connector will only detect new changes. Changes that occurred during the failover process are lost. However, initializing the `accessor.state` file presents its own problems. If synchronization is started immediately after installation to produce the `accessor.state` file, and then stopped, when synchronization is started after failing over, the Directory Server Connector will try to process all retro changelog entries. To strike a balance between these two options, the retro changelog Plug-in on the failover installation's preferred master (`master3-eu.gt.com`) is configured to only store changes for one day (this limit is increased during the failover process). When synchronization is started for the failover installation, the Directory Server Connector only processes one day's changes in the directory server.

The *Sun Java System Directory Server Enterprise Edition 6.0 Administration Guide* contains instructions on how to enable trimming of the retro changelog. Essentially, the `nsslapd-changelogmaxage` attribute in the `cn=Retro Changelog Plugin`, `cn=plugins`, `cn=config` entry is modified and Directory Server is restarted. Setting the value of this attribute to `1d` trims entries that are older than one day. The value can be modified from the command line by using `ldapmodify` or by directly editing the Directory Server's `dse.ldif` file while Directory Server is not running.

Active Directory Connector

To allow Active Directory to detect changes that occurred while it was stopped, the Active Directory Connector persistently stores the `usnChanged` value of the last change that it processed. When Active Directory begins processing changes again, it only needs to examine entries with an `usnChanged` value larger than the previous one that it has processed. Like the Directory Server Connector, the Active Directory Connector first stores this `usnChanged` high-water mark when synchronization is started for the first time. Therefore, as part of preparing the failover installation, synchronization is started to allow the Active Directory Connector to check-point its state.

When failing over and synchronization is restarted for the failover installation, the Active Directory Connector will process all entries that were modified since it was last started. Even though this will be thousands of entries, the processing will be fast because the Active Directory object cache database is up-to-date with most of the changes. Only the changes made since the last time that the `idsync resync` command was run will be processed.

Initializing the Connector State

To initialize the connector state for the failover configuration, synchronization must be started. Before synchronization can be started, the Identity Synchronization for Windows Plug-in must be enabled on both `master3-eu.gt.com` and `master4-eu.gt.com` to point to the failover configuration. After the plug-in has been enabled and the directory servers have been restarted, synchronization can be started. Verify that both connectors have entered the SYNCING state, by using the console or the `idsync printstat` command:

```
bash-2.05# ./idsync printstat -w <password omitted> -q <password omitted>
Exploring status of connectors, please wait
Connector ID: CNN100
    Type: Sun Java(TM) System Directory
    Manages: dc=gt,dc=com (ldaps://master3-eu.gt.com:636)
            (ldaps://master4-eu.gt.com:636)
    State: SYNCING
    Installed on: connectors-eu.gt.com
    Plugin SUBC100 is installed on ldaps://master3-eu.gt.com:636
    Plugin SUBC101 is installed on ldaps://master4-eu.gt.com:636
```

```
Connector ID: CNN101
    Type: Active Directory
    Manages: gt.com (ldaps://ad1-us.gt.com:636) (ldaps://ad2-us.gt.com:636)
            (ldaps://ad4-eu.gt.com:636) (ldaps://ad3-eu.gt.com:636)
    State: SYNCING
    Installed on: connectors-eu.gt.com
```

```
Sun Java(TM) System Message Queue Status: Started
```

```
Checking the System Manager status over the Sun Java(TM) System Message Queue.
```

```
System Manager Status: Started
SUCCESS
```

After synchronization has started, modify a user password in both Active Directory and Directory Server, which will force the connectors to persist their state. To verify, do the following:

- **Directory Server Connector.** On this connector, check for the presence of the `/var/opt/SUNWsw/persist/ADP100/accessor.state` file. Check that the `highestacknowledgedchangenumber` value stored in the file is not `-1`. (To determine the appropriate ADP subdirectory of `persist`, find the connector ID using the console or `idsync printstat`, then replace `CNN` with `ADP` in the connector ID.)
- **Active Directory Connector.** On this connector, check that it actually propagated the change. An `INFO` message that includes the `usnchanged` value should appear in the central log, for example:

```
[05/Nov/2004:14:07:38.982 -0600] INFO      18  CNN101 connectors-eu
"The agent is sending the following inbound action to MQ:
Type: MODIFY SUL: GT_USERS {Data Attrs: } {Other Attrs: cn: Jane Test
dn: CN=Jane Test,CN=Users,DC=gt,DC=com objectclass: top,person,
organizationalPerson, user dspswuserlink: Rwyr9YEFk0WYxbFP5Nnrjg== pwdlastset:
127441696561778218 samaccountname: 3aa00test100001 sn: test100001 usnchanged: 120831
whnchanged: 20041105230736.0Z passwordchanged: TRUE}."
(Action ID=CNN102-1000A5846CB-5, SN=2)
```

After you have verified that both connectors have check-pointed their state, stop synchronization for the failover installation. Then, reinstall the Directory Server Plug-ins on master3-eu.gt.com and master4-eu.gt.com to point to the primary configuration.

Failover Installation Maintenance

After the connectors' state has been persisted, little maintenance is done on the failover installation. `idsync resync -u` should be run periodically as described in [“Periodic idsync resync Operation for Failover Installation” on page 108](#).

Note – If any additional attributes are added to the list of synchronized attributes, a full `idsync resync -u` operation should be executed, not an incremental one. If this is not done, when the Active Directory Connector is started after a failover, it will incorrectly detect modifications to the added attribute because its object cache database did not store the previous value.

When to Fail over

Identity Synchronization for Windows is a background system that with one exception is not user-facing. Therefore, if it is temporarily unavailable, for example, due to routine hardware maintenance, most users will be unaffected. After the system is restored, Identity Synchronization for Windows will synchronize all changes that were made while it was unavailable.

The user-facing aspect is the on-demand password synchronization performed from the Directory Server Plug-in to Active Directory. If on-demand password synchronization fails, the user will not be able to log in to Directory Server. Therefore, Identity Synchronization for Windows provides more availability options for this situation. The Directory Server Plug-in can be configured to authenticate to any Active Directory domain controller. Thus, even if all but one Active Directory domain controller is down, on-demand password synchronization will still succeed.

Note – The Directory Server Plug-ins receive their configuration from the Directory Server Connector over an encrypted channel. This configuration, which includes the location of the Active Directory domain controllers and credentials, is cached in memory by the plug-in. So even if the Directory Server Connector is unavailable, it will still be able to connect to Active Directory. However, if Directory Server is restarted, the plug-in's cached configuration is lost, and on-demand synchronization on that Directory Server will fail until the Directory Server Connector is available.

Depending on the size of the deployment, the failover procedure might take minutes to over an hour to perform. Therefore, the failover procedure should not be undertaken if the Identity Synchronization for Windows outage is expected to be short and temporary, for example, during the system restart of the system that contains the Identity Synchronization for Windows Core. Use the failover procedure only in situations where Identity Synchronization for Windows must be completely reinstalled or a complete idsync resync operation must be run on a large population.

Such situations might include the following:

- Any machine that runs the Identity Synchronization for Windows Core or Connector experiences a hardware failure.
- The configuration directory that stores the Identity Synchronization for Windows configuration is corrupt.
- The Active Directory domain controller that the Active Directory Connector communicates with experiences a hardware fault and must be rebuilt.
- The preferred Directory Server is corrupted and must be initialized from another preferred Directory Server.

Failing Over

The failover process, at a high-level, involves only these tasks:

- [“Stopping Synchronization at the Primary Installation” on page 113](#)
- [“Starting Synchronization at the Failover Installation” on page 114](#)
- [“Re-enabling the Directory Server Plug-Ins” on page 114](#)
- [“Changing the PDC FSMO Role Owner” on page 115](#)
- [“Monitoring the Logs” on page 115](#)

Stopping Synchronization at the Primary Installation

Before starting synchronization at the failover installation, stop synchronization at the primary installation to prevent unwanted interaction between the two systems. Depending on the reasons for failing over, this is accomplished in different ways. If the primary installation of

Identity Synchronization for Windows is still operating properly, for example, failing over because a domain controller or Directory Server is down, stop synchronization using the console or `idsync stopsync`. Otherwise, stop the Identity Synchronization for Windows daemon (on Solaris) or service (on Windows) on each system where Identity Synchronization for Windows is still operational.

Starting Synchronization at the Failover Installation

After synchronization is stopped at the primary installation, it must be started at the failover installation by using the console or `idsync startsync`.

Note – The Directory Server Connector will not enter the SYNCING state until the Directory Server Plug-ins are reinstalled on the preferred and secondary Directory Servers (`master3-eu.gt.com` and `master4-eu.gt.com`).

The Active Directory Connector will need to process every entry in Active Directory that has been modified since it was last started. It might take several minutes for the Active Directory Connector to begin propagating changes to Directory Server. Setting the log level to INFO before starting synchronization can reduce the impact of the Active Directory Connector having to catch up.

Re-enabling the Directory Server Plug-Ins

To complete the failover process, the Directory Server Plug-in is re-enabled on each Directory Server, which ensures the following:

- The plug-ins running on the preferred Directory Servers use the encryption key from the failover installation to encrypt password changes.
- All directory servers receive an updated on-demand password synchronization configuration.
- Logging done by the plug-ins is sent to the Central Logger of the failover installation.

The plug-ins must be re-enabled in this order:

1. Failover installation's preferred Directory Server.
2. Failover installation's secondary Directory Server.
3. All other preferred and secondary Directory Servers.
4. All preferred and secondary Directory Server replicas.

Note – The order in which the Directory Server Plug-ins are enabled is important. If they are enabled in the wrong order, on-demand synchronization requests could loop between two preferred Directory Servers, tying up all Directory Server connections.

When re-enabling the plug-ins, make sure to specify the configuration directory of the failover installation, for example, `config-eu.gt.com`.

This re-enabling procedure can be automated by doing more work ahead of time:

1. Install the Directory Server Plug-ins for the failover configuration.
2. Export the plug-ins' configuration for each master from the `cn=pswsync,cn=plugins,cn=config` tree.
3. Re-enable the Directory Server Plug-ins for the primary configuration.

To fail over:

1. Delete the `cn=pswsync,cn=plugins,cn=config` tree.
2. Add the failover installation entries by using `ldapmodify`.
3. Restart the directory server.

Changing the PDC FSMO Role Owner

This task is optional. If the Active Directory Connector in the failover installation is configured to communicate with a domain controller that does not have the PDC FSMO role, synchronization from Active Directory will be delayed due to the Active Directory replication latency. To avoid this delay, the PDC FSMO role can be transferred to the domain controller that the connector is accessing.

Monitoring the Logs

After the failover process is complete, monitor the central error log of the failover installation for any unexpected warnings. Warnings similar to the following will likely appear:

```
[08/Nov/2004:07:58:24.803 -0600] WARNING 25 CNN100 connectors-eu
"Unable to obtain password of user CN=Jane Test,OU=people,DC=gt,DC=com,
because the password was encoded by a previous installation of
Identity Synchronization for Windows Directory Server
Plugin. The password of this user cannot be synchronized at this time.
Update the password of this user again in Directory Server."
```

These warnings occur for each password update in the retro changelog that was made before the Directory Server Plug-in was reinstalled because the Primary Directory Server Plug-in was configured to use a different encryption key from the failover Directory Server Plug-in. Many of

these password updates were likely synchronized by the primary installation before it went offline, but those that occurred after the primary installation went offline cannot be recovered. Users who are affected must change their password either in Active Directory or Directory Server to synchronize their passwords.

Failing Back to the Primary Installation

The procedure for failing back to the primary installation is identical.

Pluggable Authentication Modules

This appendix explains how to configure Sun Java System Identity Synchronization for Windows 6.0 and pluggable authentication modules (PAM) so that an LDAP store can provide synchronization capabilities between Solaris and Windows systems.

This appendix covers the following topics:

- “Advantages of Combining PAM and Identity Synchronization for Windows” on page 117
- “Configuring PAM and Identity Synchronization for Windows” on page 119
- “Configuring Systems to Prevent Eavesdropping” on page 137
- “Introducing Windows NT Into the Configuration” on page 137
- “Example /etc/pam.conf File” on page 138

Note – In this appendix, *Windows* refers to Windows systems using Active Directory for authentication. Windows NT systems may impose different approaches.

Advantages of Combining PAM and Identity Synchronization for Windows

If your enterprise contains both Solaris and Windows systems, you can simplify the administration of the user community if you use Identity Synchronization for Windows to manage the two sets of users as a single set.

Combining PAM and Identity Synchronization for Windows can accomplish the following goals:

- Enable an LDAP store to provide synchronization capabilities between Solaris and Windows

For example, you can enable user information (including passwords) created or modified on one system (Solaris or Windows) to replicate to its counterpart so either system can act on the information.

- Authenticate to the Solaris OS and manage passwords against an LDAP store
 - Enable users to change their own passwords, if doing so does not contradict security policy
- Configure your systems to ensure that passwords are never sent over a medium that permits eavesdropping

Note – The Solaris OS implementation of PAM has long-offered the ability to use an LDAP store. However, starting with the Solaris 9 OS, PAM modules are included by default, which makes it possible to use a product such as Identity Synchronization for Windows.

You can update the Solaris 8 OS to support this functionality by using Patch Number 108993 for SPARC® based systems or Patch Number 108994 for x86 based systems.

While some Solaris software PAM modules are LDAP-aware, other modules do not use LDAP in a way that triggers Identity Synchronization for Windows interception actions.

For example, when you configure the PAM_UNIX module to use LDAP (using a directive specified in the `/etc/nsswitch.conf` file), the module never binds (as the user in question) to the LDAP store when authenticating. Instead, the PAM_UNIX module reads the user's LDAP entry, internally compares the password found on the LDAP entry to the password provided, and makes its authentication decision accordingly.

Because the PAM_UNIX module authentication is done outside the purview of the LDAP store, so none of the hooks put into place by Identity Synchronization for Windows will be used. Consequently, passwords will fail to replicate from the LDAP store to Windows.

To initiate the synchronization process discussed in this appendix, Identity Synchronization for Windows requires that all authentication systems bind to the LDAP store. Furthermore, the binding mechanism must present the user's password in a clear manner, such as a simple bind, which rules out the use of the Simple Authentication and Security Layer (SASL) and Digest mechanisms. Using Transport Layer Security (TLS) for the connection between PAM and the LDAP store makes the use of simple binds acceptable for security.

The PAM_UNIX module's authentication methods should suffice in environments where passwords never change or where password changes always flow from the LDAP store to Windows. However, you must not use the PAM_UNIX module in environments where passwords change on Windows.

In contrast to the PAM_UNIX module, the PAM_LDAP module binds to the LDAP store using a preformed, “user-centric” distinguished name (DN) and a user-provided password when authenticating. This binding action allows Identity Synchronization for Windows to maintain the synchronization of an entry. Thus, you will use the PAM_LDAP module in conjunction with Identity Synchronization for Windows and existing PAM modules.

The following section explains how to configure PAM and Identity Synchronization for Windows.

Configuring PAM and Identity Synchronization for Windows

Note – In this section, *Windows* refers to Windows systems using Active Directory for authentication. Windows NT systems might require different approaches.

To configure Identity Synchronization for Windows and PAM for environments in which passwords can change on Windows, involves completing the following tasks:

1. [“To Configure an LDAP Repository for PAM” on page 119](#)
2. [“To Install and Configuring Identity Synchronization for Windows” on page 122](#)
3. [“To Populate the LDAP Repository” on page 123](#)
4. [“To Configure a Solaris System to Use PAM” on page 126](#)
5. [“To Verify That PAM Is Interoperating With the LDAP Store” on page 131](#)
6. [“To Demonstrate That User Changes Flow to the Reciprocal Environment” on page 132](#)

The remainder of this section provides procedures for each task and includes examples to illustrate the configuration process.

To Configure an LDAP Repository for PAM

This procedure describes how to configure an Identity Synchronization for Windows-supported LDAP repository for PAM, using the following example information:

- The LDAP store is Directory Server 6.0 software that is hosted on a Solaris Operating System.
- The host machine’s DNS name is `LDAPHOST.EXAMPLE.COM`.
- The machine’s IP address is `192.168.220.219` in the test environment.

In this example, the IP address has a concrete value so that when you configure the PAM clients, you can use the repository’s IP address to avoid potential conflicts based on how the PAM client machine resolves its DNS queries.

Prerequisites to configure an Identity Synchronization for Windows-supported LDAP repository for PAM.

- Consult the *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide* to verify that you are using a supported Directory Server.
- For PAM to work with Directory Server 6.0, you must edit the `/usr/lib/ldap/idsconfig` script and change 5 to 6 in the following code:

```
if [ "${IDS_MAJVER}" != "5" ]; then
```

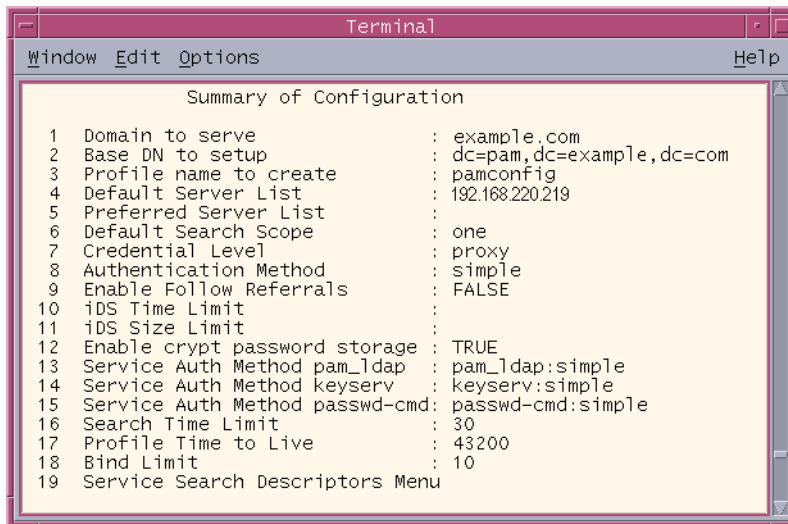
- While executing the `idsconfig` command-line tool, you need to know which values to assign to the various configuration parameters. If you do not know, use the default values when prompted (other than the configuration parameters 1, 2, and 4).

Use the following steps to configure an Identity Synchronization for Windows- supported LDAP repository for PAM.

1. Configure the LDAP store by using the Solaris OS `idsconfig` command-line tool.

The `idsconfig` tool prompts you for values that are needed to form the directory information tree (DIT) to be contained in the LDAP store. The `idsconfig` tool will manipulate the requisite LDAP store schema to accommodate the impending user population.

When you configure the test system, the following `idsconfig` summary screen is displayed:



2. To change the value of a configuration parameter, type its associated configuration number.
3. Select an option from the list of predefined options that can be supplied to the selected parameter.
4. Evaluate the following key parameters' values:
 - Domain to serve
 - Base DN to setup
 - Profile name to create
 - Service Auth Method `pam_ldap`

If necessary, use the `idsconfig` tool to change the context of these parameter values so they are appropriate for your deployment. If you are working in a test environment where you can change DNS entries and set machine IP addresses to arbitrary values, you may use the names and addresses provided in this appendix.

5. Continue with the proxy creation initiated by the `idsconfig` tool by providing the appropriate values (default or custom) for the various parameters.
6. After the configuration is complete and `idsconfig` stores the generated configuration, create virtual list view (VLV) indexes when prompted.

Note – VLV indexes (also called *browsing indexes*) enable PAM to quickly search for groups, users, and so forth. For information about creating VLV indexes, go to:

“Managing Browsing Indexes” in *Sun Java System Directory Server Enterprise Edition 6.0 Administration Guide*

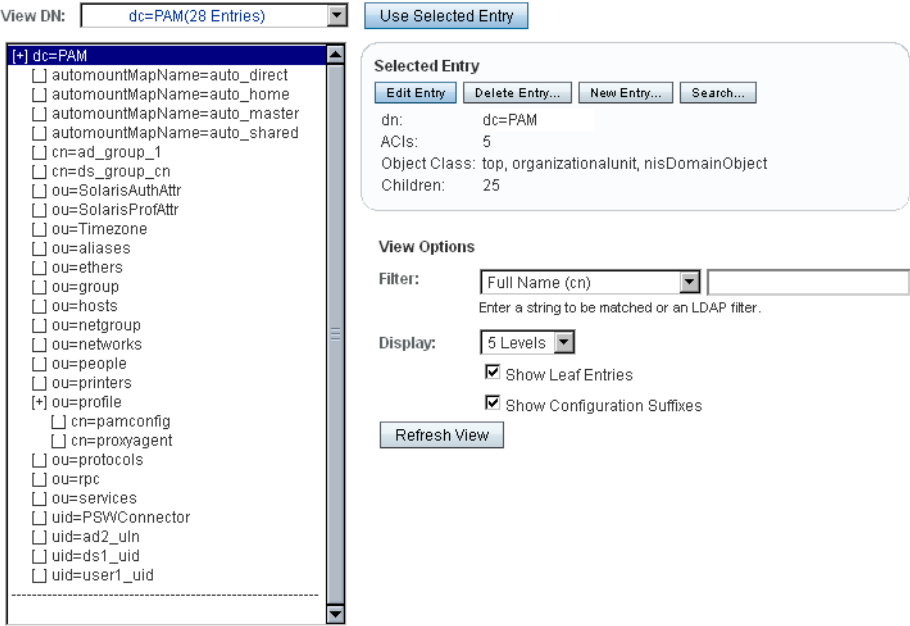
Pay particular attention to the number of VLV indexes that you are prompted to create. The `idsconfig` tool will provide a list of VLV indexes that are contextually sensitive to the state in which it finds the LDAP store.

The following figure shows the resulting topology, as displayed on the Sun Java System Directory Server Console.



Example.com:1389 - Browse Data

Choose a View DN to browse through your LDAP data. You can select an entry in the list and set it as the View DN in order to hide upper level (



When you are finished configuring the LDAP repository for PAM, continue to [“To Populate the LDAP Repository”](#) on page 123.

To Install and Configuring Identity Synchronization for Windows

This procedure starts the process of bridging the LDAP store with the Windows authentication system. Install and configure Identity Synchronization for Windows on the following two systems:

- LDAPHOST.EXAMPLE.COM
- WINHOST.EXAMPLE.COM.

You can install Identity Synchronization for Windows on the Solaris host (LDAPHOST.EXAMPLE.COM) and then configure the software so that all of the distributed processes required by Identity Synchronization for Windows will run on LDAPHOST.EXAMPLE.COM.

Instructions for installing and configuring Identity Synchronization for Windows are provided in the *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide*.

When you finish configuring Identity Synchronization for Windows, continue to [“To Populate the LDAP Repository” on page 123](#).

To Populate the LDAP Repository

After configuring an LDAP repository for PAM, you *push* user entries to the LDAP store.

For example, you create a new, single user named *George Washington* that is subordinate to the following entry:

```
ou=people,dc=pam,dc=example,dc=com
```

In addition, you use an `ou=people` container that is subordinate to the base DN you provided to `idsconfig`. You might have to make contextual changes to the base DN you are going to use.

1. In the Directory Service Control Center Console, click the Entry Management tab, and then the Browse tab. The various entity management controls are displayed in the right pane.
2. Click New Entry to display the New Entry screen.
3. Type a value in the Entry Parent DN field to specify the location to save the entity in Directory Server and click Next.

Java™ System Directory Service Control Center

New Entry

Steps Help Specify Entry Location

1. Specify Entry Location

2. Choose Object Class

3. Configure Attributes

4. Summary

Provide the location DN where the entry will be written in the Directory Server.

* Indicates required field

*Entry Parent DN:

Previous Next Cancel

4. Associate your entity with an object class by choosing an option from the Entry Type drop-down menu and press Next.

Java™ System Directory Service Control Center

New Entry

Steps Help Choose Object Class

1. Specify Entry Location

2. Choose Object Class

3. Configure Attributes

4. Summary

Choose the object type of the entry you want to create.

Entry Type:

User - (inetOrgPerson)

Static Group - (groupOfUniqueNames)

Dynamic Group - (groupOfUris)

Certificates Group - (groupOfCertificates)

Managed Role - (nsManagedRoleDefinition)

Filtered Role - (nsFilteredRoleDefinition)

Domain Component - (domain)

Organizational Unit - (organizationalUnit)

Referral - (referral)

Class of Service - (cosPointerDefinition)

User defined Object Classes

DUACConfigProfile

NisKeyObject

Previous Next Cancel

Based on the object class that you associated with your entity, a number of different attributes are displayed.

Java™ System Directory Service Control Center

New Entry

Steps

Help

1. Specify Entry Location

2. Choose Object Class

➤ 3. Configure Attributes

4. Summary

Configure Attributes

Enter the attribute values for the new entry. For multi-valued attributes, press the Enter key in the field to make the field ta

* Indicates required field

Required Attributes

Naming Attribute:

User ID (uid)

* Full Name (cn):

George

* Last Name (sn):

Washington

Allowed Attributes

First Name (givenname):

George Washington

User ID (uid):

gwashing

Password (userPassword):

Confirm Password:

E-mail (mail):

gwashing@example.com

Telephone Number (telephoneNumber):

555-555-1212

Fax Number (facsimileTelephoneNumber):

Locality (l):

Organization (o):

Organizational Unit (ou):

audio:

businessCategory:

carLicense:

departmentNumber:

description:

destinationIndicator:

displayName:

employeeNumber:

employeeType:

Previous

Next

5. Enter the appropriate values for the parameters and press Next.
A summary of the entity is displayed.

The screenshot shows the 'Java™ System Directory Service Control Center' window. The 'New Entry' section has three tabs: 'Steps', 'Help', and 'Summary'. The 'Summary' tab is active, displaying a list of steps on the left and a summary of the entry on the right. The steps are: 1. Specify Entry Location, 2. Choose Object Class, 3. Configure Attributes, and 4. Summary (highlighted with a blue arrow). The summary on the right lists the following details: Entry DN: uid=gwashing,ou=people; Object Class: User - (inetOrgPerson); Full Name (cn): George; Last Name (sn): Washington; First Name (givenname): George Washington; E-mail (mail): gwashing@example.com; Telephone Number (telephoneNumber): 555-555-1212; User ID (uid): gwashing; and Password (userPassword): *****. At the bottom of the window are 'Previous', 'Finish', and 'Cancel' buttons.

Steps	Summary
1. Specify Entry Location	Review your settings and click finish if they are correct.
2. Choose Object Class	Entry DN: uid=gwashing,ou=people
3. Configure Attributes	Object Class: User - (inetOrgPerson)
→ 4. Summary	Full Name (cn): George
	Last Name (sn): Washington
	First Name (givenname): George Washington
	E-mail (mail): gwashing@example.com
	Telephone Number (telephoneNumber): 555-555-1212
	User ID (uid): gwashing
	Password (userPassword): *****

6. Verify that the new user (George Washington) is displayed in the console.

7. Click Finish.

PAM clients can now authenticate against (and change the password for) this entry.

To Configure a Solaris System to Use PAM

After configuring the LDAP store, you must configure a Solaris system and create a PAM client to test the viability of PAM-based authentication as follows.

1. Install and configure a test Solaris system.
2. Configure the PAM client.
3. Specify new rules for authentication and password management.

To illustrate this process, example instructions and guidelines are provided in the next three sections.

Installing and Configuring a Solaris Test System

Install and configure a test Solaris system on an independent, stand-alone machine.

Note – To simplify this example, consider configuring a system that is devoid of any naming service directives (such as NIS or NIS+).

Consider using a Solaris 9 4/04 x86 based system, which contains patches required for PAM and its associated subsystems.

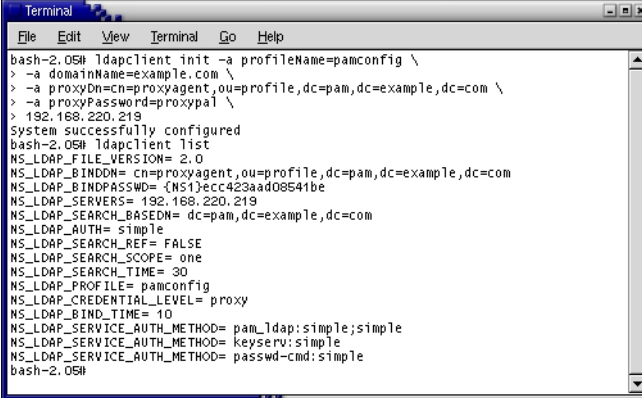
Configuring the PAM Client

The following example instructions assume that you have installed and configured the Solaris test system as described in the previous section.

You must configure a PAM client to locate the LDAP host with a repository that the client will use to access (and effectively change) the LDAP store. To configure the PAM client, use the Solaris OS `ldapclient` command, which stores the client's configuration information on the local host.

Note – Make a backup copy of the `/etc/nsswitch.conf` file before you run the `ldapclient` command. Running `ldapclient` has several side effects, which includes completely replacing the system's `/etc/nsswitch.conf` file with a copy of the content in `/etc/nsswitch.ldap`.

The following screen illustrates an example `ldapclient` command:



```

Terminal
File Edit View Terminal Go Help
bash-2.05# ldapclient init -a profileName=pamconfig \
> -a domainName=example.com \
> -a proxyDn=cn=proxyagent,ou=profile,dc=pam,dc=example,dc=com \
> -a proxyPassword=proxypal \
> 192.168.220.219
System successfully configured
bash-2.05# ldapclient list
NS_LDAP_FILE_VERSION= 2.0
NS_LDAP_BINDDN= cn=proxyagent,ou=profile,dc=pam,dc=example,dc=com
NS_LDAP_BINDPASSWD= {NS1}ecc423aad08541be
NS_LDAP_SERVERS= 192.168.220.219
NS_LDAP_SEARCH_BASEDN= dc=pam,dc=example,dc=com
NS_LDAP_AUTH= simple
NS_LDAP_SEARCH_REF= FALSE
NS_LDAP_SEARCH_SCOPE= one
NS_LDAP_SEARCH_TIME= 30
NS_LDAP_PROFILE= pamconfig
NS_LDAP_CREDENTIAL_LEVEL= proxy
NS_LDAP_BIND_TIME= 10
NS_LDAP_SERVICE_AUTH_METHOD= pam_ldap:simple:simple
NS_LDAP_SERVICE_AUTH_METHOD= keyserv:simple
NS_LDAP_SERVICE_AUTH_METHOD= passwd-cmd:simple
bash-2.05#
  
```

Other guidelines include the following:

- Use an IP address for this configuration instead of a DNS name, because DNS might not be available when the PAM client needs it.
- Use a proxy credential set to prevent anonymous authenticators from manipulating data in the LDAP store.

The system provides a set of proxy credentials that you can use when manipulating PAM data on the host LDAP store. These proxy credentials match those created when you used the `idsconfig` command to initialize the LDAP store.

The generated configuration stores the proxy's password as an encrypted value, which is done for security purposes.

In addition to generating the requisite LDAP contact information, running `ldapclient` replaces the contents of the `/etc/nsswitch.conf` file (the file you backed up earlier) with a copy of the contents in `/etc/nsswitch.ldap`. Consequently, most (or all) of the directives found in `/etc/nsswitch.conf` will include the LDAP directive, which means that the LDAP store will be consulted when resolving the associated service request.

In this example, the resulting `/etc/nsswitch.conf` file left on the system by the `ldapclient` command dropped the DNS directive from the list of used services when resolving hosts. As the example LDAP store might not be populated with the requisite host information needed to supplant DNS, the `/etc/nsswitch.conf` file is adjusted. In this example, this is the only change made to the post `ldapclient` command version of the `/etc/nsswitch.conf` file.

Edit the host's declaration to read as follows:

```
hosts: files ldap dns
```

Do not use the following reconfigured value (using `ldapclient`):

```
hosts: ldap [NOTFOUND=return] files
```

This adjustment might not address your environment's needs correctly if you are running your DNS from the LDAP store. Only apply this change if your environment's context depends on it. In addition, continue to compare and contrast the service directives with the effective `/etc/nsswitch.conf` file to the pre-`ldapclient` variant to validate that all services are now being directed correctly.

Specifying Rules for Authentication and Password Management

Note – The example instructions provided in this section assume that you completed the tasks as described in [“Installing and Configuring a Solaris Test System” on page 126](#).

When you configure a Solaris system to use PAM, change the `/etc/pam.conf` file to incorporate the new rules that you want it to use for authentication and password management. For an example, see [“Example /etc/pam.conf File” on page 138](#).

Before making any changes to the `/etc/pam.conf` file, make sure that you make a backup copy of the original `/etc/pam.conf` file. Changes made to the `/etc/nsswitch.conf` and `/etc/pam.conf` files can render your PAM client inaccessible, which means that your configuration will deny everyone's (including root) authentication access to the machine.

If you need to recover from a situation of this type, do the following:

1. Edit the `pam.conf` file in the current session. See [“Authentication” on page 129](#) and [“Password Management” on page 131](#).
2. In a new terminal window, try connecting to the local host using the `rsh` or `ssh` command and then try logging in.
 - If you fail to authenticate, you can still correct the problem using the open window that you used in the previous step.
 - If you are still unable to recover, restore the `/etc/nsswitch.conf` and `/etc/pam.conf` files to their original state.

Using the Solaris OS `sys-unconfig` command might not restore your system because this command does not affect the `/etc/nsswitch.conf` and `/etc/pam.conf` files.
3. Repeat steps 1 and 2 until you achieve the expected system behavior.

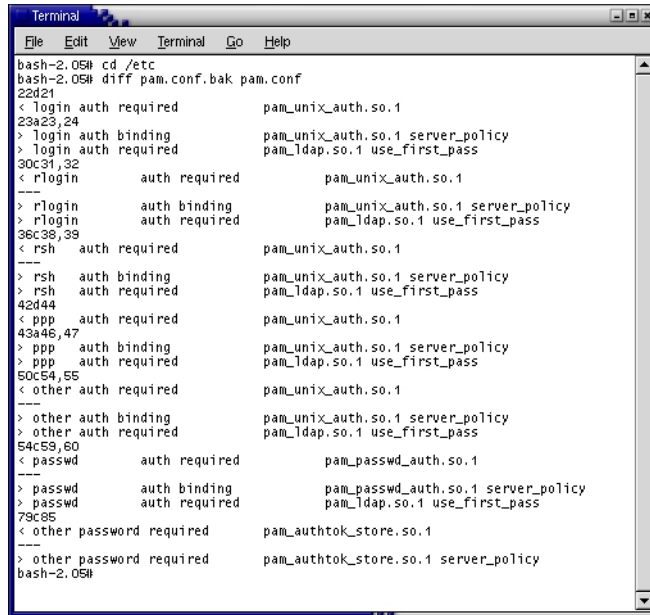
The changes you must make to `/etc/pam.conf` are minor, but important, and are explained in the next two sections:

Authentication

For purposes of authentication, you must edit the `/etc/pam.conf` file as follows:

1. Locate any entries in the original `/etc/pam.conf` file that direct the system to use a rule requiring `PAM_UNIX_AUTH`, and edit them to accept a binding directive and to pass the `server_policy` parameter to the `PAM_UNIX_AUTH` module.

The following output shows a `diff` between the original `/etc/pam.conf` file and the edited file.



```

Terminal
File Edit View Terminal Go Help
bash-2.05# cd /etc
bash-2.05# diff pam.conf.bak pam.conf
22d21
< login auth required          pam_unix_auth.so.1
23a23,24
> login auth binding          pam_unix_auth.so.1 server_policy
> login auth required          pam_ldap.so.1 use_first_pass
30c31,32
< rlogin      auth required      pam_unix_auth.so.1
-----
> rlogin      auth binding        pam_unix_auth.so.1 server_policy
> rlogin      auth required        pam_ldap.so.1 use_first_pass
36c38,39
< rsh      auth required          pam_unix_auth.so.1
-----
> rsh      auth binding            pam_unix_auth.so.1 server_policy
> rsh      auth required            pam_ldap.so.1 use_first_pass
42d44
< ppp      auth required          pam_unix_auth.so.1
-----
> ppp      auth binding            pam_unix_auth.so.1 server_policy
> ppp      auth required            pam_ldap.so.1 use_first_pass
50c54,55
< other auth required            pam_unix_auth.so.1
-----
> other auth binding              pam_unix_auth.so.1 server_policy
> other auth required              pam_ldap.so.1 use_first_pass
54c58,60
< passwd      auth required          pam_passwd_auth.so.1
-----
> passwd      auth binding            pam_passwd_auth.so.1 server_policy
> passwd      auth required            pam_ldap.so.1 use_first_pass
79c85
< other password required          pam_authtok_store.so.1
-----
> other password required          pam_authtok_store.so.1 server_policy
bash-2.05#

```

2. Edit the file to add a new rule after the altered rule line. The `/etc/pam.conf` file is processed from the top down so line order is important.

The new rule requires the service to include PAM_LDAP when determining whether to accept an authentication request. The `use_first_pass` parameter tells the PAM_LDAP module that it must accept a password collected by an earlier rule's module (usually satisfied by the PAM_AUTHTOK_GET module).

Note – A use case that deserves special consideration is how PAM treats the login of a local user. A *local user* is a user who is permitted by `/etc/nsswitch.conf` directives to examine files (such as the root account) *and* is listed in the `/etc/passwd` file. Local users are not necessarily stored in the LDAP store.

Allowing the root user to be listed in the LDAP store would simplify management of an important user account that spans the topology. However, an equally powerful case could be made for systems whose root user must be kept “private” for a given machine.

To accommodate the need to keep an account (such as root) as a local user, PAM must be configured so it does not access the LDAP back-end store if the user information has been saved in the local files. This situation can be addressed by specifying the `server_policy` parameter for the PAM_UNIX_AUTH module in the `/etc/pam.conf` configuration file.

Password Management

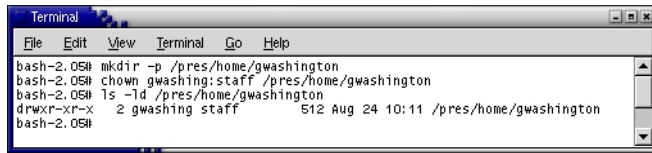
The only effective change required for password management is to append the `server_policy` parameter to the `PAM_AUTHTOK_STORE` module. When you use the `server_policy` parameter, the module will update passwords for local users (if found) or access the LDAP store accordingly. If the module cannot find a user either locally or in the LDAP store, the system will provide an appropriate error message.

When you have finished configuring the Solaris test machine, continue to [“To Verify That PAM Is Interoperating With the LDAP Store” on page 131](#).

To Verify That PAM Is Interoperating With the LDAP Store

You now test whether the newly configured Solaris system can operate as a PAM client.

1. At the command line, create George default home directory, `/pres/home/gwashington`, on your test system.



```
Terminal
File Edit View Terminal Go Help
bash-2.05# mkdir -p /pres/home/gwashington
bash-2.05# chown gwashing:staff /pres/home/gwashington
bash-2.05# ls -ld /pres/home/gwashington
drwxr-xr-x  2 gwashing staff      512 Aug 24 10:11 /pres/home/gwashington
bash-2.05#
```

The PAM client is working because `gwashing` is both understood and displayed.

2. Configure the `auto_home` system on which to mount that file system automatically. The following output shows that the PAM LDAP client system that you configured can authenticate as `gwashing`. In addition, this output demonstrates that a password can be changed and that the new password will be accepted on a subsequent authentication request.

```

Terminal
File Edit View Terminal Go Help
bash-2.05$ hostname
pamclient.example.com
bash-2.05$ rsh -l gwashing pamclient.example.com
Password:
Last login: Tue Aug 24 10:38:48 from pamclient.examp
Sun Microsystems Inc. SunOS 5.9 Generic January 2003
bash-2.05$ passwd
passwd: Changing password for gwashing
Enter existing login password:
New Password:
Re-enter new Password:
passwd: password successfully changed for gwashing
bash-2.05$ exit
logout
Connection to pamclient.example.com closed.
bash-2.05$ rsh -l gwashing pamclient.example.com
Password:
Last login: Tue Aug 24 10:51:11 from pamclient.examp
Sun Microsystems Inc. SunOS 5.9 Generic January 2003
bash-2.05$ exit
logout
Connection to pamclient.example.com closed.
bash-2.05$

```

3. Check the LDAP store log for non-search operations. An audit of the LDAP operations completed in support of the preceding login and password-change test is displayed.

```

Terminal
File Edit View Terminal Go Help
bash-2.03$ hostname
ldaphost
bash-2.03$ pwd
/var/mps/serverroot/slapd-ldaphost/logs
bash-2.03$ cat access | grep gwashing | grep -v SRCH
[24/Aug/2004:10:51:11 -0500] conn=8 op=0 nsid=1 - BIND dn="uid=gwashing,ou=people,dc=pam,dc=example,dc=com" method=128 version=3
[24/Aug/2004:10:51:11 -0500] conn=8 op=0 nsid=1 - RESULT err=0 tag=97 nentries=0 etime=0 dn="uid=gwashing,ou=people,dc=pam,dc=example,dc=com"
[24/Aug/2004:10:51:16 -0500] conn=20 op=0 nsid=1 - BIND dn="uid=gwashing,ou=people,dc=pam,dc=example,dc=com" method=128 version=3
[24/Aug/2004:10:51:16 -0500] conn=20 op=0 nsid=1 - RESULT err=0 tag=97 nentries=0 etime=0 dn="uid=gwashing,ou=people,dc=pam,dc=example,dc=com"
[24/Aug/2004:10:51:21 -0500] conn=29 op=0 nsid=1 - BIND dn="uid=gwashing,ou=people,dc=pam,dc=example,dc=com" method=128 version=3
[24/Aug/2004:10:51:21 -0500] conn=29 op=0 nsid=1 - RESULT err=0 tag=97 nentries=0 etime=0 dn="uid=gwashing,ou=people,dc=pam,dc=example,dc=com"
[24/Aug/2004:10:51:21 -0500] conn=29 op=1 nsid=2 - MOD dn="uid=gwashing,ou=people,dc=pam,dc=example,dc=com"
[24/Aug/2004:10:51:28 -0500] conn=37 op=0 nsid=1 - BIND dn="uid=gwashing,ou=people,dc=pam,dc=example,dc=com" method=128 version=3
[24/Aug/2004:10:51:28 -0500] conn=37 op=0 nsid=1 - RESULT err=0 tag=97 nentries=0 etime=0 dn="uid=gwashing,ou=people,dc=pam,dc=example,dc=com"
bash-2.03$

```

To Demonstrate That User Changes Flow to the Reciprocal Environment

Both Windows and the LDAP store (if so configured) use a one-way hash when storing passwords. This configuration prevents true replication of password data between the two systems, but does not prevent the password synchronization.

For an environment that is participating in bidirectional password synchronization, any existing user's entry being tracked in both environments must be in one of the following states:

- Both Windows and the LDAP store contain current information.
No work needs to be done on either system because the entries are current.
- Windows is current but the LDAP store is outdated or stale.

The LDAP store is in a state of waiting. Specifically, the system is waiting for an opportunity to capture the current, correct password for the LDAP store. In this case, properly configured Identity Synchronization for Windows software marks the corresponding entry in the LDAP store as outdated or stale.

When a system with Directory Server and Identity Synchronization for Windows gets a bind to a stale user, Identity Synchronization for Windows replays the given bind credentials to the configured Windows Active Directory to see if the user-supplied password is acceptable. If Active Directory authenticates the password, Identity Synchronization for Windows modifies the LDAP store so that it now possesses the password. Directory Server might eventually hash the value as part of its password policy and clear the stale status.

This process is known as *on-demand synchronization*, which results in both systems becoming synchronized with regard to the given entry after a successful bind occurs to the LDAP store.

- Windows is stale but the LDAP store is current.

This state can occur under normal circumstances, but most frequently occurs due to one of the following situations:

- When the LDAP store possesses all the entries and Windows is waiting to be populated by Directory Server entries

This situation requires the most attention to get the system into a synchronized state because existing entries in the LDAP store presumably have one-way, hashed passwords. When Identity Synchronization for Windows tries to migrate data to Windows, there are no usable values to push when propagating the user's password. In this case, Windows will request a password when a user logs in for the first time.

If multiple users logging in to your Windows systems without an effective password is not acceptable, you can set a password programmatically for each entry in question. After Identity Synchronization for Windows reports that the system is synchronizing, you can use LDAP operations to specify the generated passwords on the LDAP store side or the Windows side of your environment.

To complete this process, you must specify a must-change password policy after the user's next logon. This policy facilitates the transformation of an end user's password from an openly known value to a secret value.

- When an end user changes his or her password on the LDAP store side and Identity Synchronization for Windows is propagating the change through the system (either actively or by scheduling the work)

In this situation, the information on Windows remains stale for just a short time. The Identity Synchronization for Windows components have captured all of the information needed to bring the Windows system into a synchronized state. Identity Synchronization for Windows takes the time only to process the information, then the entry will be synchronized.

- Both Windows and the LDAP store are stale.

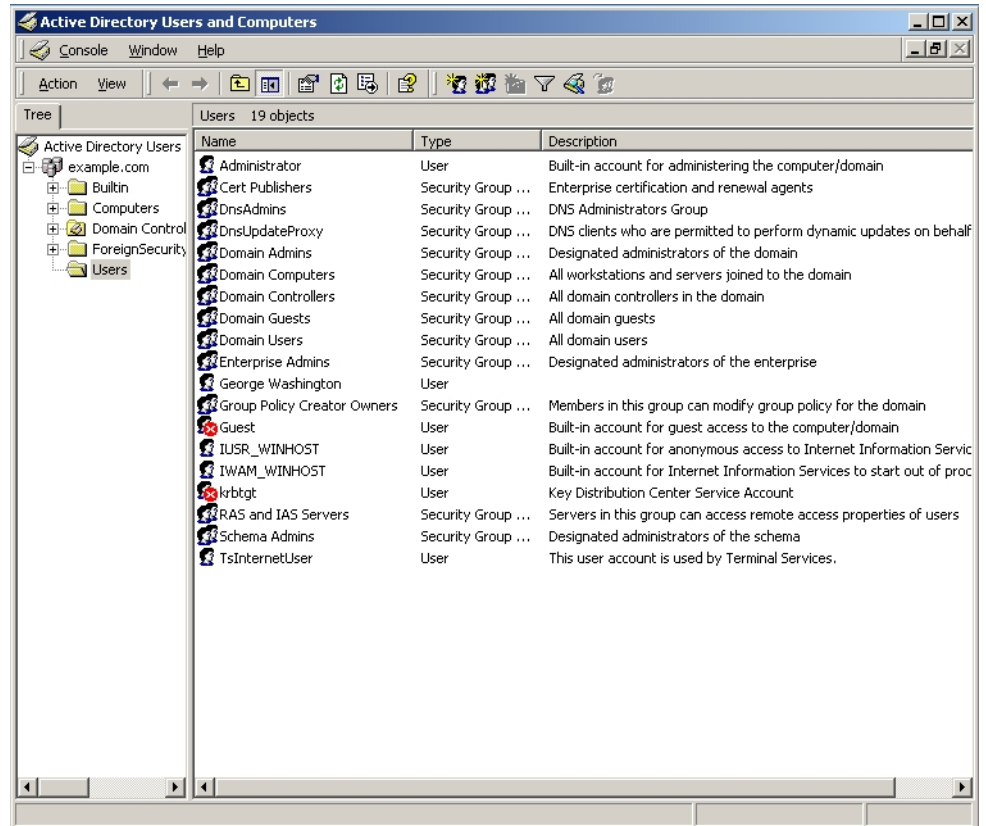
If both systems are stale, you cannot ascertain which system contains the authoritative information. You must decide where to place the true state of an entry into one of the three, previously mentioned states. For example, moving the entry from your LDAP store creates corresponds to the previous state.

Note – This situation can lead to a very tedious process as you might have to examine every user entry.

If you created a user called *George Washington* on a Solaris system that operates as a PAM client, and then use the `idsync resync` command to push the entry to Windows, you can verify that Identity Synchronization for Windows has also created the entry on Windows as explained in the following procedure.

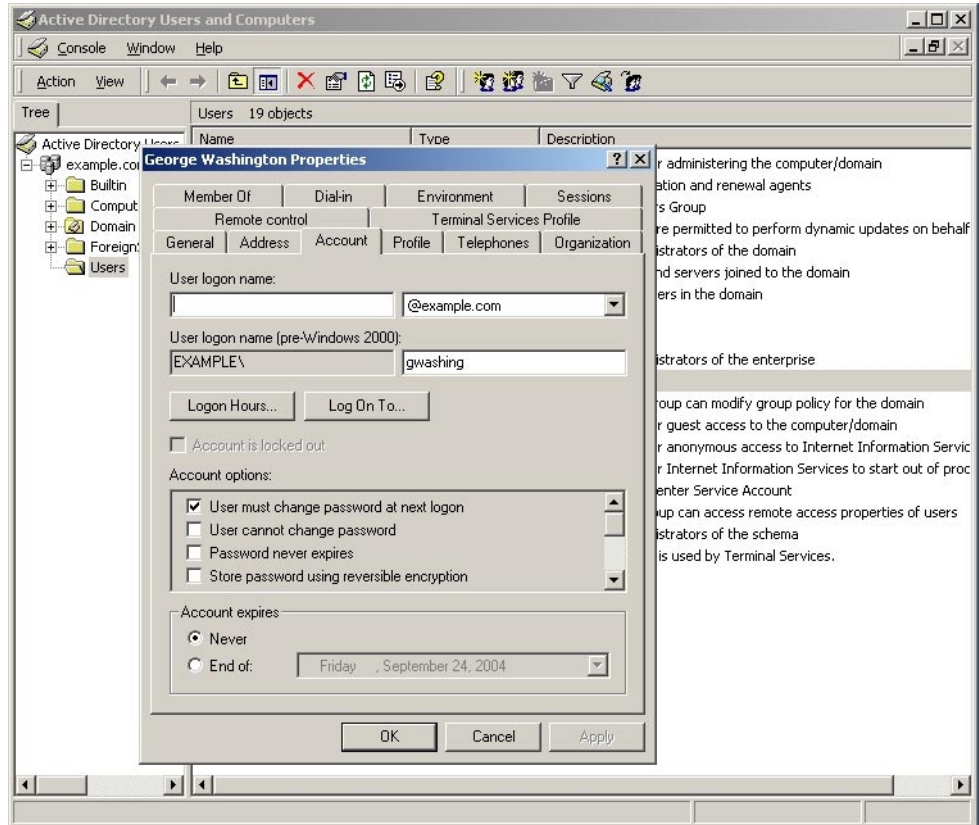
▼ **To Verify Entries on Windows**

- 1 From the Windows Start menu, go to Control Panel → Administrative Tools → Active Directory User and Computers.
- 2 When the Active Directory User and Computers window is displayed, go the Active Directory Users pane (on the left) and click Users.



3 Right-click the George Washington entry, and choose Properties.

When the George Washington Properties dialog box is displayed, look at the Account options section. It shows that the User Must Change Password at Next Logon check box is selected, which means that George Washington will be required to change his password the next time that he logs in.



4 Log in as George Washington.

Windows is correctly tracking the entry because the log-in attempt displays the Logon Message dialog box stating, “Your password has expired and must be changed.”

5 Click OK to close the Logon Message dialog box and to display the Change Password dialog box to provide a new password.

6 Type and confirm a new password, but *do not* provide a value for the Old Password field.

This is first time the user has logged in (since being created over protocol), so supplying an old password value will cause an error message and Windows will ask you to enter the new password again.

7 Click OK to save the new password and close the Change Password dialog box.

If Windows accepts the new password, a message is displayed stating that the new password has been accepted.

At this point, the George Washington entry has moved from where the Windows entry is stale and the LDAP store is current to where Windows is current and the LDAP store entry is stale.

George Washington's entry will maintain this condition until the next time that it binds to the LDAP store. At that time, the entry will move to where the entry is current on both Windows and the LDAP store.

Configuring Systems to Prevent Eavesdropping

This appendix does not include procedure for configuring systems so that communication between systems is always conducted securely to prevent eavesdropping.

Some of the required configuration changes are addressed when you configure Identity Synchronization for Windows. For example, starting with Windows 2000, the password policies require that all password changes be made using secure methods. Consequently, simply configuring the Windows system partially addresses the security requirement.

However, eavesdroppers can still see the bind attempts when Identity Synchronization for Windows components replay bind credentials. To address this issue, you *must* configure Identity Synchronization for Windows to communicate securely with its Windows data source by configuring the Identity Synchronization for Windows Connectors to trust certificates offered by the Windows Active Directory system.

In addition, you must ensure that all clients authenticating to the LDAP store do so over TLS. You must configure PAM clients to trust the LDAP store and ensure that `idsconfig` specifies `TLS:pam_ldap:simple` as the only authentication method for the LDAP store.

The root accounts cannot use the `passwd` command arbitrarily to change a user's password on PAM client. You might consider this restriction to be a limitation, depending on whether you trust the PAM client administrators.

Introducing Windows NT Into the Configuration

While the term *Windows* refers to Windows platforms using Active Directory for authentication, the system being discussed in this appendix can use Windows NT in place of (or along with) newer variations of Windows.

Be aware, however, that Windows NT lacks the ability to use on-demand synchronization normally provided by Identity Synchronization for Windows.

The Identity Synchronization for Windows on-demand synchronization process must be able to bind to Windows over LDAP, with a set of candidate credentials, an ability that the Windows NT authentication system lacks. When Identity Synchronization for Windows is configured with Windows NT, it expects password changes to be captured at their source and at the time

the change is made. This capture requirement has ramifications when you initially start a system that uses Identity Synchronization for Windows. Specifically, Identity Synchronization for Windows needs to see a password change for any entry before the entry is actually synchronized.

Synchronization in Windows NT environment involves modifying the passwords of all entries in the LDAP store using an LDAP-based utility. As these modifications go through the LDAP store system, Identity Synchronization for Windows forwards the captured passwords to the Windows NT system, and resulting in no stale passwords on these two systems. However, because you created the passwords by deterministic means, these passwords might be easy to guess.

You can limit potential security breaches if you use Windows NT password policy to force all users to change their password at their next login. As each user changes their password, the Identity Synchronization for Windows password DLL installed on the Primary Domain Controller forwards the password change to the LDAP store.

Example /etc/pam.conf File

The following /etc/pam.conf file is provided to help you configure and run Identity Synchronization for Windows and PAM.

Note – *This /etc/pam.conf file is only an example.*

The file's configuration is not appropriate for all situations. Analyze the content thoroughly before using this file in a production environment.

EXAMPLE A-1 Example /etc/pam.conf File

```
#
#ident    "@(#)pam.conf    1.20    02/01/23 SMI"
#
# Copyright 1996-2006 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
# PAM configuration
#
# Unless explicitly defined, all services use the modules
# defined in the "other" section.
#
# Modules are defined with relative pathnames, i.e., they are
# relative to /usr/lib/security/$ISA. Absolute path names, as
# present in this file in previous releases are still acceptable.
#
```

EXAMPLE A-1 Example /etc/pam.conf File (Continued)

```

# Authentication management
#
# login service (explicit because of pam_dial_auth)
#
login    auth requisite    pam_authtok_get.so.1
login    auth required     pam_dhkeys.so.1
login    auth required     pam_dial_auth.so.1
login    auth binding      pam_unix_auth.so.1 server_policy
login    auth required     pam_ldap.so.1 use_first_pass
#
# rlogin service (explicit because of pam_rhost_auth)
#
rlogin    auth sufficient   pam_rhosts_auth.so.1
rlogin    auth requisite    pam_authtok_get.so.1
rlogin    auth required     pam_dhkeys.so.1
rlogin    auth binding      pam_unix_auth.so.1 server_policy
rlogin    auth required     pam_ldap.so.1 use_first_pass
#
# rsh service (explicit because of pam_rhost_auth,
# and pam_unix_auth for meaningful pam_setcred)
#
rsh       auth sufficient   pam_rhosts_auth.so.1
rsh       auth binding      pam_unix_auth.so.1 server_policy
rsh       auth required     pam_ldap.so.1 use_first_pass
#
# PPP service (explicit because of pam_dial_auth)
#
ppp       auth requisite    pam_authtok_get.so.1
ppp       auth required     pam_dhkeys.so.1
ppp       auth required     pam_dial_auth.so.1
ppp       auth binding      pam_unix_auth.so.1 server_policy
ppp       auth required     pam_ldap.so.1 use_first_pass
#
# Default definitions for Authentication management
# Used when service name is not explicitly mentioned for authentication
#
other     auth requisite    pam_authtok_get.so.1
other     auth required     pam_dhkeys.so.1
other     auth binding      pam_unix_auth.so.1 server_policy
other     auth required     pam_ldap.so.1 use_first_pass
#
# passwd command (explicit because of a different authentication module)
#
passwd    auth binding      pam_passwd_auth.so.1 server_policy
passwd    auth required     pam_ldap.so.1 use_first_pass
#

```

EXAMPLE A-1 Example /etc/pam.conf File (Continued)

```
# cron service (explicit because of non-usage of pam_roles.so.1)
#
cron    account required    pam_projects.so.1
cron    account required    pam_unix_account.so.1
#
# Default definition for Account management
# Used when service name is not explicitly mentioned for account management
#
other    account requisite    pam_roles.so.1
other    account required    pam_projects.so.1
other    account required    pam_unix_account.so.1
#
# Default definition for Session management
# Used when service name is not explicitly mentioned for session management
#
other    session required    pam_unix_session.so.1
#
# Default definition for Password management
# Used when service name is not explicitly mentioned for password management
#
other    password required    pam_dhkeys.so.1
other    password requisite    pam_authtok_get.so.1
other    password requisite    pam_authtok_check.so.1
other    password required    pam_authtok_store.so.1 server_policy
#
# Support for Kerberos V5 authentication (uncomment to use Kerberos)
#
#rlogin    auth optional    pam_krb5.so.1 try_first_pass
#login     auth optional    pam_krb5.so.1 try_first_pass
#other     auth optional    pam_krb5.so.1 try_first_pass
#cron      account optional    pam_krb5.so.1
#other     account optional    pam_krb5.so.1
#other     session optional    pam_krb5.so.1
#other     password optional    pam_krb5.so.1 try_first_pass
```

Configuring Identity Manager and Identity Synchronization for Windows to Coexist

This appendix describes how Sun Java System Identity Synchronization for Windows 6.0 and Identity Manager 5.0 SP2 coexist in a customer deployment as part of a larger user support strategy, to facilitate native password changes on Directory Server and Active Directory.

This appendix augments the [Chapter 3](#) integration of Identity Synchronization for Windows with Identity Manager, and the changes required in Identity Manager, for coexistence.

This appendix assumes that you have knowledge of the concepts of and deployment experience with Identity Manager.

This appendix covers the following topics:

- “Components for Deploying Identity Manager and Identity Synchronization for Windows” on page 141
- “Identity Manager and Identity Synchronization for Windows Functionality” on page 143
- “Password Changes on Active Directory” on page 144
- “Password Changes on Directory Server” on page 144
- “Password Changes and Provisions Originating From Identity Manager Administrator Interface” on page 145
- “Configuring Identity Manager and Identity Synchronization for Windows” on page 145

Components for Deploying Identity Manager and Identity Synchronization for Windows

This section explains the main components of the Identity Manager and Identity Synchronization for Windows deployment:

- Active Directory domains
- Separate Directory Server deployment

- Any other Identity Manager-managed resource (which does not include the previous two) for example, Oracle RDBMS

The Identity Manager Administrator Interface handles resource administration, such as system-wide password changes and user creations. All password changes between Directory Servers and Active Directory domains are synchronized using Identity Synchronization for Windows. Password changes that occur within an Active Directory Domain are synchronized to Directory Server using Identity Synchronization for Windows, and synchronized to all other Identity Manager resources using pwsync, an Identity Manager Dynamic Link Library (DLL) installed on the Primary Domain Controllers of Windows systems. All password changes originating from the Identity Manager Administrator Interface are subsequently propagated to all Identity Manager resources, except the Sun Java System Directory Server. All user creations originating from the Identity Manager Administrator Interface are propagated to all resources, including Directory Servers. See also [“Configuring pwsync to Not Propagate Passwords to Directory Server”](#) on page 146.

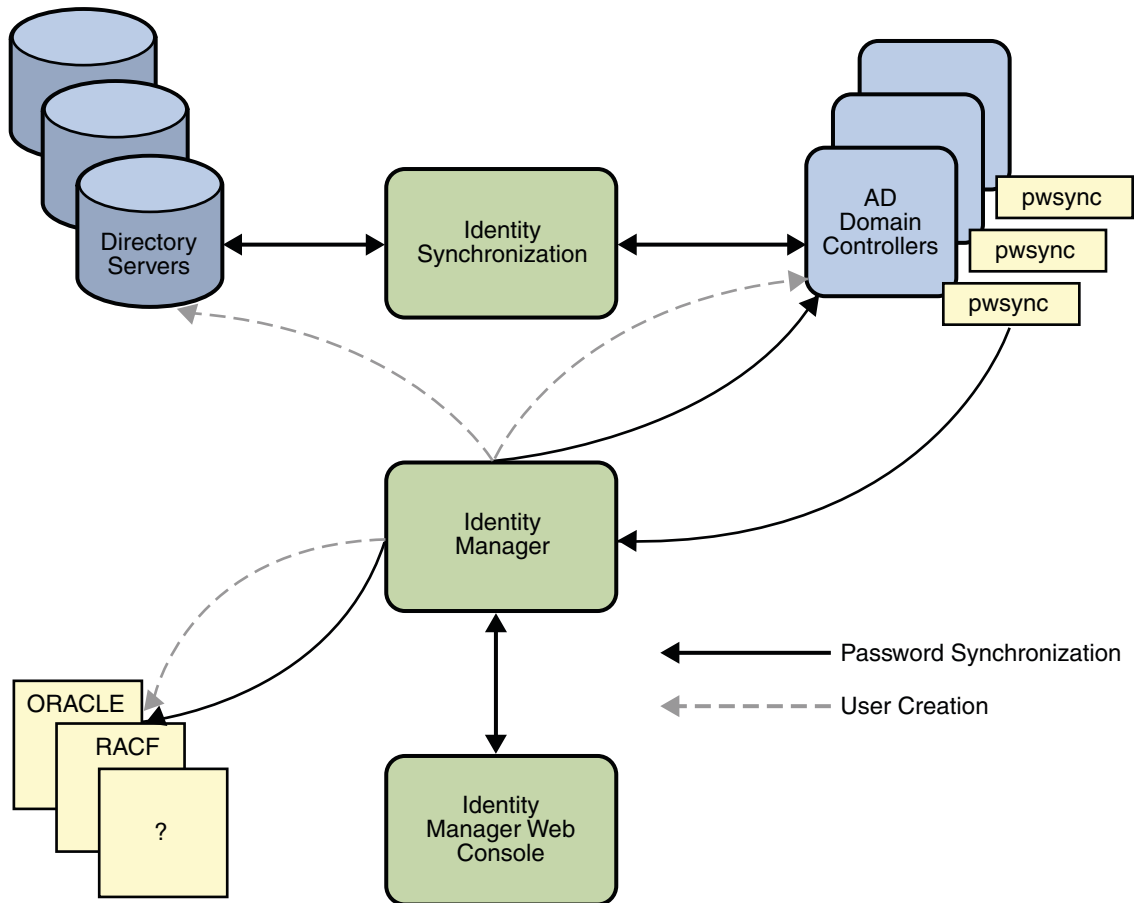


FIGURE B-1 Password Synchronization and User Creation in an Identity Manager and Identity Synchronization for Windows Environment

Identity Manager and Identity Synchronization for Windows Functionality

For Identity Manager and Identity Synchronization for Windows to work effectively, both must be deployed and configured to function as a single system.

Identity Synchronization for Windows functionality includes the following:

- Detection of all the password changes on Active Directory, and synchronization with Directory Server using on-demand synchronization
- Detection of all password changes on Directory Server, and synchronization with Active Directory

Identity Synchronization for Windows does not synchronize these functions:

- User creations
- User deletions
- Non-password attributes

The Identity Manager functionality, coexisting with Identity Synchronization for Windows, includes the following:

- Detection of all password changes on Active Directory using pwsync, and synchronization of the changes to all other Identity Manager-managed resources, except Directory Server resources
- Use of the Identity Manager Administrator Interface to propagate user password changes to Active Directory and all other Identity Manager-managed resources, except Directory Servers
- Use of the Identity Manager Administrator Interface to propagate new users across all Identity Manager-managed resources (including Directory Servers)

Password Changes on Active Directory

Passwords modified on Active Directory are propagated through the Identity Manager and Identity Synchronization for Windows deployment as described here.

1. The user resets the password on Active Directory by using the Change Password option in the Task Manager dialog box of Windows.
2. Identity Synchronization for Windows detects the change and sets a password invalid flag on the corresponding user entry in the Identity Synchronization for Windows-managed Directory Server.
3. The user connects to Directory Server for the password change to be complete (see “Using On-Demand Password Synchronization to Obtain Clear-Text Passwords” in *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide* in *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide*).
4. Identity Manager’s pwsync DLL also detects the password change and propagates it to all other Identity Manager-managed resources, except Directory Servers.

Password Changes on Directory Server

Passwords modified on Directory Servers are propagated through the Identity Manager and Identity Synchronization for Windows deployment as described here:

1. The user changes the password on Directory Server.
2. The password change is detected by Identity Synchronization for Windows and propagated to Active Directory.

3. Identity Manager's pwsync DLL also detects the password change and propagates it to all other Identity Manager-managed resources, except Directory Servers.

Password Changes and Provisions Originating From Identity Manager Administrator Interface

Password changes that occur through the Identity Manager Administrator Interface are propagated to all Identity Manager-managed resources, except Directory Server. When a password change is detected on Active Directory, Identity Synchronization for Windows synchronizes it with Directory Server.

User creation originating from the Identity Manager Administrator Interface is propagated to all Identity Manager-managed resources, including both Directory Server and Active Directory domains. New users will have to be linked by Identity Synchronization for Windows. For details, see [“Configuring pwsync to Not Propagate Passwords to Directory Server” on page 146](#) and [“Handling Identity Manager-Provisioned Users” on page 148](#).

Configuring Identity Manager and Identity Synchronization for Windows

Configuring Identity Manager and Identity Synchronization for Windows can involve the following tasks:

- Setting up Identity Manager 5.0 SP2
- Setting up Identity Manager 5.0 SP1
- Configuring Identity Synchronization for Windows

Setting Up Identity Manager 5.0 SP2

Starting with Identity Manager 5.0 SP2, a new form property was introduced that prevents the Directory Server resource from being shown as a resource where passwords can be changed. Identity Manager 5.0 SP2 also introduced a new system configuration property that can be used to prevent pwsync from propagating password changes to the Directory Server resource.

Configuring the Form Property

You must ensure that Identity Manager does *not* propagate user password changes to Directory Server, but *only* to Active Directory. You must also ensure that Identity Manager relies on Identity Synchronization for Windows to propagate the password changes to Directory Server.

To prevent a resource from being displayed in the table of resources where password changes occur, add the following form property to any form that is used for changing a user's password.

```
<Properties\>
  <Property name='Exclude'\>
    <list\>
      /<new class='com.waveset.object.AttributeCondition'\>
        <s\>id</s\>
        <s\>equals</s\>
        <s\>#ID#50D9481DC6C43026:3BB34:FFB73A9286:-7FC0</s\>
      </new\>/
    </list\>
  </Property\>
</Properties\>
```

The resource can be excluded by id (as shown in the form), by name (a string), or by type (also a string). The forms to which this property must be included are as follows:

- Change User Password Form
- Change My Password Form
- Change Password Form
- Expired Login Form
- Reset User Password Form
- Tabbed User Form

If some of these forms already include the form property, only the new attribute condition needs to be added (from the XML fragment in this procedure).

Note – In multiple attribute condition scenarios, the forms are and'ed together (they cannot be or'ed). For example, if the Change My Password Form and Change Password Form already include an attribute condition to exclude disabled resources, and the id condition is added, a resource will only be excluded if it meets both conditions, that is, it is disabled and has the ID you entered.

If a form does not already include the Exclude property, add it by copying the full XML fragment in this procedure or add the <Property name=Exclude>, if a <Properties> block already exists.

Configuring pwsync to Not Propagate Passwords to Directory Server

The passwordSyncExcludeList system configuration attribute lists resources that should not be updated when the Active Directory pwsync DLL detects a password change. In an Identity Manager and Identity Synchronization for Windows environment, this attribute should include Directory Servers that are being synchronized, to prevent unwanted interaction between Identity Manager and Identity Synchronization for Windows.

1. Go to the /debug page, for example, <http://applicationserverhost:port/idm/debug>
2. List objects of type Configuration
3. Add the following attributes to the system configuration file:

```
<Attribute name='passwordSyncExcludeList' value='Directory Server Resource'/>
```

where *Directory Server Resource* is the name of the resource to be excluded during a pwsync password change. If you need to exclude more than one resource, use a comma-separated list.)

Setting Up Identity Manager 5.0 SP1

Identity Manager installations prior to 5.0 SP2 require a modification to the workflow for coexistence with Identity Synchronization for Windows. These changes facilitate propagation of updates from other Identity Manager resources to Directory Servers (provisioning) and to Active Directory (passwords and provisioning). Administrators must install the Identity Manager component pwsync on all domain controllers where password synchronization is desired with all other Identity Manager-managed resources, except Directory Server.

This workflow change will result in an error during the end-user password change. However, the change is propagated to Directory Server.

To modify the Identity Manager workflow:

1. Install pwsync on all Active Directory domain controllers.
2. Modify the Identity Manager task definition for Change User Password from either the /debug page or the Configuration Editor.
 - a. Add a new activity to remove the Directory Server resources that should not have the password reset.
 - b. Replace the string NAME DS RESOURCE with the names of these Directory Server resources by adding the following activity:

```
<Activity id='1' name='RemoveLDAP'\>
  <Variable name='userObject'/>
  # checkout user #
  <Action id='0' application='com.waveset.session.WorkflowServices'\>
    <Argument name='op' value='checkoutObject'/>
    <Argument name='type' value='User'/>
    <Argument name='name' value='${accountId}'/>
    <Argument name='authorized' value='true'/>
    <Return from='object' to='userObject'/>
  </Action>
  <Action id='1'\>
    <expression>
      <block>
        # Get pending changes for Directory Server resource #
        <defvar name='resourceInfo'\>
          <invoke name='getResourceInfo'\>
            <ref>userObject</ref>
```

```

        <s\>NAME DS RESOURCE</s\>
    </invoke\>
</defvar\>

    # Clears pending password change #
    <invoke name='setPassword'\>
        <ref\>resourceInfo</ref\>
        <null\>
    </invoke\>
    # Get other pending resource changes #
    <defvar name='resourceInfoAttributes'\>
        <invoke name='getAttributes'\>
            <ref\>resourceInfo</ref\>
        </invoke\>
    </defvar\>
    # removes expire password flag #
    <invoke name='remove'\>
        <ref\>resourceInfoAttributes</ref\>
        <s\>expirePassword</s\>
    </invoke\>
    # Set cleared attributes for check in #
    <invoke name='setAttributes'\>
        <ref\>resourceInfo</ref\>
        <ref\>resourceInfoAttributes</ref\>
    </invoke\>
</block\>
</expression\>
</Action\>
    # Check in user #
    <Action id='2' application='com.waveset.session.WorkflowServices'\>
        <Argument name='op' value='checkinObject'\>
        <Argument name='object' value='$(userObject)'\>
    </Action\>
    <Transition to='Reprovision'\>
</Activity\>

```

Configuring Identity Synchronization for Windows

Identity Synchronization for Windows should be configured as described in [Chapter 3](#), and not for user creations or any other attribute synchronization.

Handling Identity Manager-Provisioned Users

User creation is not the responsibility of Identity Synchronization for Windows in this deployment. Therefore, new users that are added to Directory Server using Identity Manager will not be linked to the corresponding entries in Active Directory domains, or conversely. To

establish this link for new users, an administrator must periodically execute `idsync resync` so that password changes for the new entries are synchronized.

The frequency with which this operation is executed is the administrator's decision and the periodic automated execution is performed using a scheduled UNIX `cron` job. For details, see [“Periodic idsync resync Operation for Primary Installation” on page 104](#).

Using Audit and Debug Logging to Isolate Problems

This appendix expands on the logging and troubleshooting information in the *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide* by describing how to use audit and debug logging to isolate problems in the system.

This appendix covers the following topics:

- “Audit Logging and Action IDs” on page 151
- “Debug Logging” on page 158
- “Windows NT Change Detection” on page 162
- “Changing the Location of the Central Logs” on page 165
- “Password Changes and Provisions Originating From Identity Manager Administrator Interface” on page 145
- “Isolating Problems in Directory Server” on page 166
- “Isolating Problems in Message Queue” on page 167
- “Isolating Problems in Active Directory” on page 167

Audit Logging and Action IDs

Before enabling debug logging, make sure that setting the audit log level to **FINEST** does not produce the desired level of information. When a new log level setting is saved in the console, it is immediately propagated to every component, and messages at that log level and above are logged to the central log.

Note – Changes to the log level are not propagated to subcomponents, that is, the Directory Server Plug-in, the Windows NT Change Detector, and the Windows NT Password Filter DLL, until the next configuration change is saved. To propagate a new log level setting to the subcomponents, make any minor edit to the configuration, undo the edit, and then save the configuration.

Action Types

When an Identity Synchronization for Windows Connector detects a change to a user entry, it uses an action to represent this change as it travels through Identity Synchronization for Windows. Each action includes a type such as `CREATE`, `MODIFY`, or `DELETE`, and enough attributes from the user entry to allow the destination connector to synchronize the change. The type of an action appears in log entries and can be one of the following:

- **MODIFY** — Indicates that an existing user entry was modified.
- **CREATE** — Indicates that a new user entry was created.
- **DELETE** — Indicates that a user entry was deleted.
- **LINK** — When Identity Synchronization for Windows synchronizes a new Directory Server user to Windows, the Windows Connector sends a Link Action to the Directory Server Connector after the entry is created. This Link Action contains the user's Windows GUID, which is written to the user's Directory Server entry. Link Actions are only sent from a Windows Connector to the Directory Server Connector.
- **UNKNOWN** — When a Windows Connector detects a change to a user, the corresponding action is assigned the Unknown type until the connector determines the type of action by comparing it with the object cache.
- **REFRESH** — Indicates resynchronization operations resulting from running the `idsync resync` command. Actions of this type hold the current value of all user attributes and do not correspond to a detected change in a user entry.
- **SENTINEL** — Signals that all source actions are sent during the `idsync resync` operation. During `idsync resync` operation, the source connector sends one Sentinel action for each Synchronization User List (SUL) that is processed, to signal that all users in that SUL are sent. This is the only action type that does not correspond to a user entry.

Attributes from the user entry are also included when the action is logged. These attributes are divided into two types: data attributes and other attributes. Data attributes correspond to attributes that are always synchronized, for example, `userpassword`. The other attributes (sometimes referred to as meta attributes) are required, for example, `objectguid` or `pwdlastset`.

Connector Layers: Accessor, Controller, and Agent

During processing, an action is passed through three different layers of the connector: accessor, controller, and agent. Each layer has different responsibilities:

- **Accessor** — Interfaces directly with a directory source over protocols such as LDAP. There are separate accessor implementations for Directory Server, Active Directory, and Windows NT.

- **Controller** — Is a connector component that interfaces with the agent and accessor components. The controller performs key synchronization-related tasks such as determining a user's membership in a SUL, searching for and linking equivalent user entries, and detecting changes to users by comparing current user entries with the previous versions stored in the object cache.

Note that the term adapter refers to the combination of the accessor and controller layers.

- **Agent** — Interfaces with Message Queue and translates attributes between their Directory Server names and Windows names. Distributes configuration received over Message Queue to the other connector layers.

For better performance, the connector processes many actions at once, causing log messages for different actions to be interleaved. When the processing of an action is complete, the source of the action (that is, the accessor or Message Queue) is notified by an acknowledgement. After an action is acknowledged, the source of the action is no longer responsible for redelivering the action in the event of a failure. The action path that detected changes from the directory source to Message Queue is known as the *inbound path*. The action path that applies changes from Message Queue to the directory source is known as the *outbound path*. These action paths are summarized in the following figure.

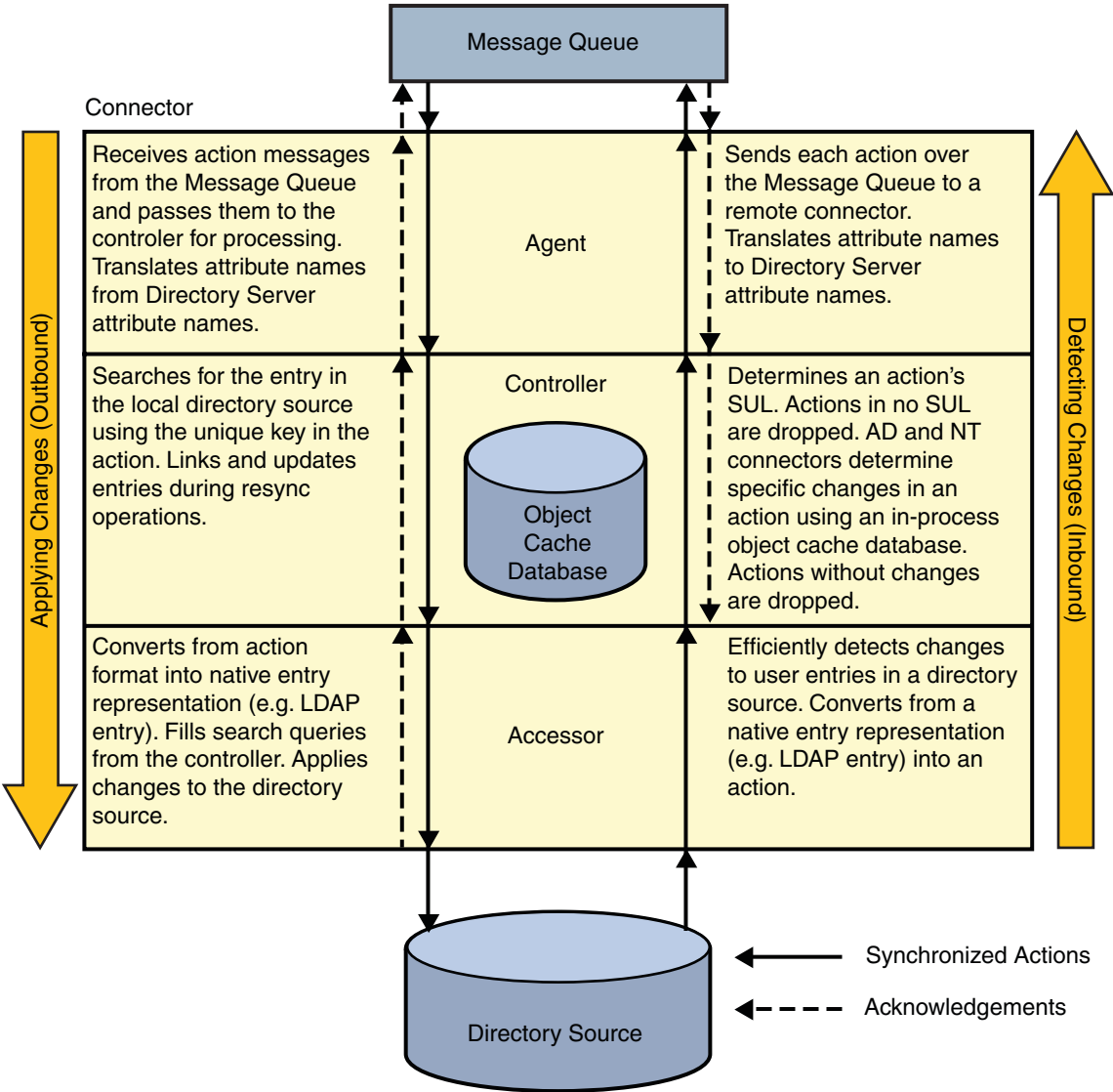


FIGURE C-1 Action Paths

To illustrate how this process applies to log messages, a sample message from the central audit.log follows and shows when user John Test's password was changed.

```
[01/Nov/2004:07:17:56.082 -0600] FINE 54 CNN101 connectors-us
"The controller has received the following inbound action from the accessor:
Type: UNKNOWN {Data Attrs: }
```

```
{Other Attrs: cn: John Test dn: CN=John Test,CN=Users,DC=gt,DC=com
objectclass: top, person, organizationalPerson,
user objectguid: coFNlF3ePUWC5Vm32GENfg== pwdlastset: 127437958757034806
samaccountname: jtest sn: Test usnchanged: 313464 whenchanged: 20041101151755.0Z}."
(Action ID=CNN101-FFEF4A8986-206, SN=1)
```

This message was logged as the accessor passed a detected change along the inbound path to the controller. The type of action is UNKNOWN because the controller has not yet compared the action with its object cache database. There are no data attributes in the entry because password values cannot be extracted from Active Directory. Instead, the `pwdlastset` attribute is used to detect if the password has changed. The action also includes a unique ID that can be used to track the action through the system. A monotonically increasing sequence number (the SN value) also appears in the log entry. In rare circumstances when messages logged at different connectors arrive out-of-order at the Central Logger, the sequence number can be used to correctly order the messages. However, sequence numbers might not always increase by one. This action can be tracked through the log by searching for the unique ID `ID=CNN101-FFEF4A8986-206`.

```
bash-2.05# grep 'ID=CNN101-FFEF4A8986-206' audit.log
[01/Nov/2004:07:17:56.082 -0600] FINE    54
CNN101 connectors-us "The controller has received the following inbound action
from the accessor: Type: UNKNOWN {Data Attrs: }
{Other Attrs: cn: John Test dn: CN=John Test,CN=Users,DC=gt,DC=com
objectclass: top, person, organizationalPerson,
user objectguid: coFNlF3ePUWC5Vm32GENfg== pwdlastset: 127437958757034806
samaccountname: jtest sn: Test usnchanged: 313464 whenchanged: 20041101151755.0Z}."
(Action ID=CNN101-FFEF4A8986-206, SN=1)
[01/Nov/2004:07:17:56.307 -0600] FINE    51
```

```
CNN101 connectors-us
"The agent has received the following inbound action from the controller:
Type: MODIFY SUL: GT_USERS {Data Attrs: }
{Other Attrs: cn: John Test dn: CN=John Test,CN=Users,DC=gt,DC=com
objectclass: top, person, organizationalPerson,
user objectguid: coFNlF3ePUWC5Vm32GENfg== pwdlastset: 127437958757034806
samaccountname: jtest sn: Test usnchanged: 313464
whenchanged: 20041101151755.0Z passwordchanged: TRUE}."
(Action ID=CNN101-FFEF4A8986-206, SN=4)
```

```
[01/Nov/2004:07:17:56.320 -0600] INFO    51
CNN101 connectors-us
"The agent is sending the following inbound action to MQ:
Type: MODIFY SUL: GT_USERS {Data Attrs: }
{Other Attrs: cn: John Test dn: CN=John Test,CN=Users,DC=gt,DC=com
objectclass: top, person, organizationalPerson,
user dspwuserlink: coFNlF3ePUWC5Vm32GENfg== pwdlastset: 127437958757034806
samaccountname: jtest sn: Test usnchanged: 313464
```

whenchanged: 20041101151755.0Z passwordchanged: TRUE}."
(Action ID=CNN101-FFEF4A8986-206, SN=6)

[01/Nov/2004:07:17:56.448 -0600] FINE 16
CNN100 connectors-us "The agent has received an outbound action from MQ:
Type: MODIFY SUL: GT_USERS {Data Attrs: }
{Other Attrs: cn: John Test dn: CN=John Test,CN=Users,DC=gt,DC=com
objectclass: top, person, organizationalPerson,
user dspswuserlink: coFNlF3ePUWC5Vm32GENfg== pwdlastset: 127437958757034806
samaccountname: jtest sn: Test usnchanged: 313464
whenchanged: 20041101151755.0Z passwordchanged: TRUE}."
(Action ID=CNN101-FFEF4A8986-206, SN=7)

[01/Nov/2004:07:17:56.462 -0600] FINE 16
CNN100 connectors-us
"The controller has received the following outbound action from the agent:
Type: MODIFY SUL: GT_USERS {Data Attrs: }
{Other Attrs: cn: John Test dn: CN=John Test,CN=Users,DC=gt,DC=com
objectclass: top, person, organizationalPerson,
user dspswuserlink: coFNlF3ePUWC5Vm32GENfg== pwdlastset: 127437958757034806
samaccountname: jtest sn: Test usnchanged: 313464
whenchanged: 20041101151755.0Z passwordchanged: TRUE}."
(Action ID=CNN101-FFEF4A8986-206, SN=8)

[01/Nov/2004:07:17:56.549 -0600] FINE 40
CNN100 connectors-us
"Applying outbound action to Accessor"
(Action ID=CNN101-FFEF4A8986-206, SN=9)

[01/Nov/2004:07:17:56.557 -0600] FINE 40
CNN100 connectors-us
"The accessor has received the following outbound action from the controller:
Type: MODIFY SUL: GT_USERS {Data Attrs: } {Other Attrs: cn: John Test
objectclass: top, person, organizationalPerson,
user dspswuserlink: coFNlF3ePUWC5Vm32GENfg== pwdlastset: 127437958757034806
samaccountname: jtest sn: Test usnchanged: 313464
whenchanged: 20041101151755.0Z passwordchanged: TRUE
dn: uid=jtest,ou=People, dc=gt,dc=com}." (Action ID=CNN101-FFEF4A8986-206, SN=10)

[01/Nov/2004:07:17:56.640 -0600] FINE 40
CNN100 connectors-us
"Successfully modified user 'uid=jtest,ou=People, dc=gt,dc=com',
action=Type: MODIFY SUL: GT_USERS {Data Attrs: [REPL dspswvalidate: true]}."
(Action ID=CNN101-FFEF4A8986-206, SN=11)

[01/Nov/2004:07:17:56.657 -0600] INFO 40
CNN100 connectors-us
"Successfully modified user uid=jtest,ou=People, dc=gt,dc=com"
(Action ID=CNN101-FFEF4A8986-206, SN=12)

Most log entries dealing with an action include the Action ID, but some log entries do not. In these cases, the log entry without the Action ID is usually close to log messages that do include the ID. For example, a summary of the LDAP modification for the previous password update appears between two other log messages:

```
[01/Nov/2004:07:17:56.557 -0600] FINE    40
CNN100 connectors-us
"The accessor has received the following outbound action from the controller:
Type: MODIFY SUL: GT_USERS {Data Attrs: }
{Other Attrs: cn: John Test objectclass: top, person, organizationalPerson,
user dspswuserlink: coFNlF3ePUWC5Vm32GENfg== pwdlastset: 127437958757034806
samaccountname: jtest sn: Test usnchanged: 313464
whnchanged: 20041101151755.0Z passwordchanged:
TRUE dn: uid=jtest,ou=People, dc=gt,dc=com}."
(Action ID=CNN101-FFEF4A8986-206, SN=10)

[01/Nov/2004:07:17:56.567 -0600] FINE    40
CNN100 connectors-us "LDAP Modify Request:
[REPLACE dspswvalidate: true] [REPLACE dspswloop: true] [REPLACE dspswloop: ] "

[01/Nov/2004:07:17:56.640 -0600] FINE    40
CNN100 connectors-us
"Successfully modified user 'uid=jtest,ou=People, dc=gt,dc=com', action=Type:
MODIFY SUL: GT_USERS {Data Attrs: [REPL dspswvalidate: true]}."
(Action ID=CNN101-FFEF4A8986-206, SN=11)
```

Note – All messages that appear in the error log file also appear in the audit log file, to facilitate correlation of events leading up to the error or warning message.

The central log files are an aggregation of the individual component logs. As long as a connector can access Message Queue, all messages written to the local logs will also be written to the central log.

A special central log file is used for idsync resync operations. The information in this log is only available on the Core machine in the logs/central/resync.log file. During a resynchronization operation, some information is also logged to the audit log, so the Action ID might be required to correlate log entries between the two files.

Directory Server Plug-In

Increasing the global log level alone might not display all the log entries from the Directory Server Plug-in. Editing Directory Server's cn=config entry might also be required. To enable more informational logging, set the following attributes in Directory Server's cn=config entry.

- nsslapd-info-log-area: 65536

- `nsslapd-infolog-level: 0`

This entry can be manipulated over LDAP while Directory Server is running, or by editing the `dse.ldif` file when Directory Server is not running.

Note – `nsslapd-infolog-area` is a bit-mask entry. If it already has a value, bit-wise OR in the 16th least-significant bit into the value.

To enable additional logging, set the attributes in Directory Server's `cn=config` entry as follows:

- `nsslapd-infolog-area: 65536`
- `nsslapd-infolog-level: 1`

Debug Logging

Most of the Identity Synchronization for Windows components have debug logging capability. This section describes how to enable debug logging for each component.

Note – Using debug logging to isolate a problem can be a time-consuming process. Be sure to read Chapter 12, “Troubleshooting,” in *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide* in *Sun Java System Directory Server Enterprise Edition 6.0 Installation Guide* before using this method.

Debug Logging in Java Components

To enable debug logging in one of the Java components (connector, System Manager, or Central Logger), edit the process's command line in the `WatchList.properties` file to include the `-Dcom.sun.directory.wps.flags.Flags.DBG=true` flag, and restart the Identity Synchronization for Windows daemon (on Solaris) or service (on Windows). For example, debug logging for the CNN101 Connector has been enabled in the following example. The existing entry for the CNN101 Connector in the `/var/opt/SUNWiw/resources/WatchList.properties` file is shown here.

```
process.name[2]=CNN101
process.command[2]=/usr/java/bin/java -Xmx256m -Xrs -DimqConnectionType=TLS
-Djava.library.path=/opt/SUNWiw/lib:/usr/lib/mps/secv1
-Djava.util.logging.config.file=/var/opt/SUNWiw/resources/Log.properties
-Dcom.sun.directory.wps.logging.directory=/var/opt/SUNWiw/logs/CNN101
-DPSWHOME=/var/opt/SUNWiw -DWPCNFG=resources
-classpath /opt/SUNWiw/lib/common.jar:/opt/SUNWiw/lib/connector.jar:

/opt/SUNWiw/lib/db.jar:/opt/SUNWiw/lib/ldapjdk.jar:
```

```

/opt/SUNWiw/lib/manager.jar:
/opt/SUNWiw/lib/registry.jar:
/opt/SUNWiw/lib/watchdog.jar:
/var/opt/SUNWiw/resources:/opt/SUNWiw/locale/resources:
/usr/share/lib/jms.jar:/usr/share/lib/imq.jar:
/usr/share/lib/mps/secv1/jss3.jar:
/usr/sfw/share/lib/xerces-200.jar:. com.sun.directory.wps.controller.
AgentHarness CNN101

process.delay[2]=120000
process.interval[2]=60000

```

Note – The `idsync printstat` command provides information about the Connector ID and the installation location, which can be used to find the correct entry in the `WatchList.properties` list.

In the following example, the command-line entry for this connector has been edited to include the special debug option. It is safest to include this option as the first option for the Virtual Machine for the Java Platform (JVM™ tool interface).

```

process.name[2]=CNN101
process.command[2]=/usr/java/bin/java
-Dcom.sun.directory.wps.flags.Flags.DBG=true
-Xmx256m -Xrs -DimqConnectionType=TLS
-Djava.library.path=/opt/SUNWiw/lib:/usr/lib/mps/secv1
-Djava.util.logging.config.file=/var/opt/SUNWiw/resources/Log.properties
-Dcom.sun.directory.wps.logging.directory=/var/opt/SUNWiw/logs/CNN101
-DPSWHOME=/var/opt/SUNWiw -DWPSCNFG=resources
-classpath /opt/SUNWiw/lib/common.jar:/opt/SUNWiw/lib/connector.jar:
/opt/SUNWiw/lib/db.jar:/opt/SUNWiw/lib/ldapjdk.jar:
/opt/SUNWiw/lib/manager.jar:/opt/SUNWiw/lib/registry.jar:
/opt/SUNWiw/lib/watchdog.jar:/var/opt/SUNWiw/resources:
/opt/SUNWiw/locale/resources:/usr/share/lib/jms.jar:
/usr/share/lib/imq.jar:/usr/share/lib/mps/secv1/jss3.jar:
/usr/sfw/share/lib/xerces-200.jar:.
com.sun.directory.wps.controller.AgentHarness CNN101
process.delay[2]=120000
process.interval[2]=60000

```

After enabling this option, stop and start the Identity Synchronization for Windows daemon (on Solaris) or service (on Windows) so that the changes take effect.

To prevent conflicts with Message Queue, wait 30 seconds after stopping the Identity Synchronization for Windows daemon or service before restarting it. When the process starts, it will write three new logs, `logs/CNN101/debug.log`, `logs/CNN101/debugErrors.log`, and `logs/CNN101/resyncDebug.log`.

- `debug.log`— Includes all debug log messages, as well as all log messages from the audit log file.
- `debugErrors.log`— Includes all debug, warning, and error messages, as well as all messages from the error log file.
- `resyncDebug.log`— Includes all resynchronization log messages that are normally only sent to the central log.

Enabling debug logging has an impact on performance and security. Debug logging can generate trace-level information that consumes more disk space than audit logs, requiring additional processor cycles that can reduce throughput. Although no sensitive information is ever written to the audit log, the debug log might include sensitive information such as passwords.

Unlike audit logging, the amount of information in the debug log is not controlled by the global log level in the console. Instead, debug logging is controlled by the `Log.properties` file located in the `resources/` directory. The primary settings that can be changed in this file are the log levels. The log levels for debug logging behave identically to the settings for the audit logs but give more fine-grained control.

The `com.sun.directory.wps.logging.debugLogger.loggerLevel = FINE` line in `Log.properties` sets the default log level to `FINE`, but individual components change the log level to increase or decrease the default amount of logging. In general, the defaults provided in this file will produce an adequate amount of debug logging without populating the log with unnecessary information.

The following table summarizes the component-level debug log levels. In the Component column, `com.sun.directory.wps.logging.DebugLogger.prefix` is implied:

TABLE C-1 Component-Level Debug Log Levels

Component	Type of Logging	Log.properties Level
<code>accessor.level</code>	Interaction with directory sources for detecting and applying changes, which is useful when problems accessing a directory source need to be diagnosed.	<code>FINER</code>
<code>accessor.saint.level</code>	Communication between the connector and the subcomponents.	<code>FINE</code> (if not specified, inherits from <code>accessor.level</code>)
<code>controller.level</code>	Processing that occurs within the controller, including determining membership in an SUL and interaction with the object cache.	<code>FINER</code>
<code>agent.level</code>	Processing that occurs within the agent, including mapping attributes, sending messages over Message Queue, and receiving messages from Message Queue.	<code>FINER</code>

TABLE C-1 Component-Level Debug Log Levels (Continued)

Component	Type of Logging	Log.properties Level
agentout.level	Processing that occurs within the agent on actions that are received from Message Queue.	<default\>
encryption.level	Encryption and decryption routines.	INFO
mq.level	Interaction with Message Queue.	<default\>
manager.level	Processing done by the System Manager.	INFO (increasing to FINE+ can significantly slow performance during resynchronization operations)
centralLogger	Processing done by the Central Logger.	<default\> (can only be set in CentralLoggerLog.properties)
crum.level	Processing done by the configuration system that retrieves, manages, validates, and saves configuration in the configuration directory.	INFO
common.level	Processing done by common utility components.	INFO
tasks.level	Low-level execution information for each thread in the system.	INFO (increasing to FINE+ can significantly slow performance)
xml.level	Low-level parsing details for XML configuration objects.	INFO (increasing to FINE+ can significantly slow performance)

These log levels can be changed by editing the Log.properties files. The changes will be reflected after a restart.

Note – All messages that appear in the audit log file also appear in the debug log file to facilitate correlation of events between the logs.

Debug Logging in the Installer

The installer and uninstaller can be configured to write extra debugging information to the installer log file, for example,

Identity_Synchronization_for_Windows_install-20041025035143.log, by setting the ISW_DEBUG_INSTALL environment variable to true before starting the installer.

- On Solaris systems, the installer log files are written to the /var/sadm/install/logs directory.

- On Windows systems, these log files are written to a temporary directory, which is controlled with the %TEMP% environment variable.

For example:

```
bash-2.05# export ISW_DEBUG_INSTALL=true
bash-2.05# ./runInstaller.sh
```

Note – Secure information such as passwords might appear in the installer log file when debug logging is enabled.

Debug Logging in the Console

The console logs some information to the central log, but most information is only logged if the console is started with the `-D` option to enable debug logging. The `-D` option accepts a single argument that controls the amount of logging to generate. The argument varies from 1 (least) to 9 (most). By default, the logging information is only written to `stderr`, but it can also be redirected to a file using the `-f` option, for example:

```
bash-2.05# ./startconsole -D 7 -f /tmp/console.log
```

Windows NT Change Detection

The first time that you start the Windows NT Connector, with no existing object cache present, the corresponding Java process's memory usage is approximately 26 – 30 Mbytes. However, the Windows NT Connector's memory usage can easily reach 100 Mbytes during normal operations.

The Windows NT Connector periodically polls the Windows NT subcomponents, that is, the Identity Synchronization for Windows NT Change Detector and the Identity Synchronization for Windows Password Filter DLL for changes. After the connector retrieves changes from a subcomponent, the connector processes the change and based on the configuration, it will or will not send the change over Message Queue. After the connector finishes processing the change, it removes the change from the subcomponents.

If the Windows NT Connector is unavailable for a longer period of time, the subcomponents can accumulate a significant number of changes in memory. A typical change record requires less than 100 bytes of memory, so 10 Mbytes of memory can accommodate 100,000 changes. However, this scenario is not likely to occur on a Windows NT system.

Both the Change Detector service and the Password Filter DLL service maintain some internal counters. You can use these counters (with the connector's log file) to determine how far behind the connector is with change processing compared to the subcomponents.

For example, the following is a user creation entry in the connector's log:

```
[01/Nov/2004:10:29:07.390 -0600]
INFO    18 CNN102 example-wnt
"The agent is sending the following inbound action to MQ:
Type: CREATE SUL: NT {Data Attrs: [ADD userpassword: ****]}
{Other Attrs: sn: test cn: Jane Test user_rid: 1004
dspswuserlink: RID=1004+DOMAIN=EXAMPLE subcomp_src:
PASSWORD_FILTER usnchanged: 5}."
(Action ID=CNN102-FFF4EFF25B-995, SN=0)
```

The important fields in this entry are:

- **subcomp_src: PASSWORD_FILTER** — Indicates that the change was read from the Identity Synchronization for Windows Password Filter DLL.
- **usnchanged: 5** — Indicates that the Password Filter DLL assigned change number 5 to this record.

If you open the registry editor and navigate to HKEY_LOCAL_MACHINE\\SOFTWARE\\Sun Microsystems\\Sun Java System Identity Synchronization for Windows\\1.1, you can view the following registry values:

```
"PfLastAackedChangeNumber"=hex:05,00,00,00,00,00,00,00
"PfLastAackedChangeTime"=hex:54,64,86,41
"PfHighestChangeNumber"=hex:05,00,00,00,00,00,00,00
"PfHighestChangeTime"=hex:52,64,86,41
```

- **PfLastAackedChangeNumber** — Specifies the last change number the connector acknowledged to the Password Filter.
- **PfLastAackedChangeTime** — Specifies what time the last update was received, in the following format:
number of seconds since 00:00:00 UTC, January 1, 1970
- **PfHighestChangenumberr** — Specifies the highest change number recorded by the Password Filter DLL. If this change number is the same as the **PfLastAackedChangeNumber**, there are no changes buffered in the Password Filter DLL.
- **PfHighestChangeTimer** — Specifies what time the change was processed by the Password Filter DLL.

The following is another example from the Identity Synchronization for Windows Change Detector:

```
[01/Nov/2004:10:30:31.802 -0600]
INFO    18 CNN102 example-wnt
"The agent is sending the following inbound action to MQ:
Type: MODIFY SUL: NT {Data Attrs: }
{Other Attrs: sn: test cn: Jane Test user_rid: 1004
```

```
dspswuserlink: RID=1004+DOMAIN=EXAMPLE subcomp_src:
NT_CHANGE_DETECTOR usnchanged: 19}."
(Action ID=CN102-FFF4EFF25B-1162, SN=0)
```

In this example, subcomp_src is set to NT_CHANGE_DETECTOR, and the following counters are displayed in the same registry entry mentioned previously:

```
"LastProcessedSecLogRecordNumber"=dword:0000001b
"LastProcessedSecLogTimeStamp"=dword:418664a7
"HighestChangeNumber"=dword:00000013
```

In this example, the Change Detector does not show the last acknowledged changenumber, but the value of HighestChangeNumber is the same as the value in the log entry.

To determine (approximately) how many unprocessed changes there are in the Change Detector, check the last log entry in which subcomp_src is set to NT_CHANGE_DETECTOR, and then compare the usnchanged value in the log entry with the HighestChangeNumber in the registry.

The LastProcessedSecLog counter refers to the last-processed entry in the system's security log.

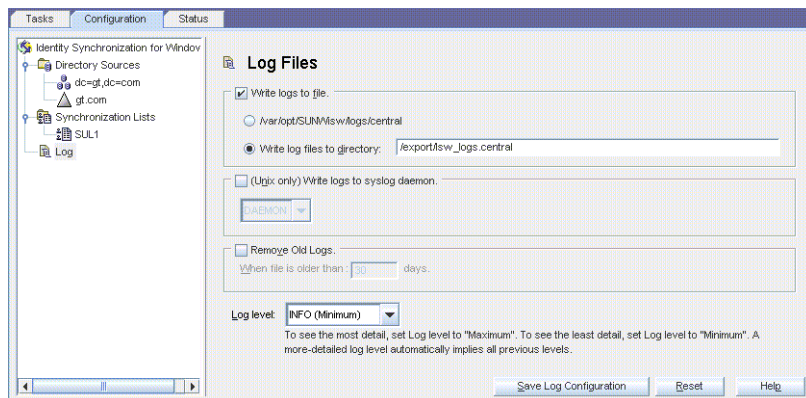
The psloglist.exe file in the PsTools software bundle at <http://www.sysinternals.com/> facilitates listing the contents of the security log and printing out the security log numbers. For example:

```
C:\>psloglist.exe security
Security log on \\.\ EXAMPLE-WNT:
[027] Security
      Type: AUDIT SUCCESS
      Computer: EXAMPLE-WNT
      Time: 11/1/04 10:30:31 AM   ID: 642
      User: Administrator\EXAMPLE
User Account Changed:
      Target Account Name: test
      Target Domain: EXAMPLE
      Target Account ID:
S-1-5-21-975783841-454902410-2021868755-1004
      Caller User Name: Administrator
      Caller Domain: EXAMPLE
      Caller Logon ID: (0x0,0x144B)
      Privileges:
...
```

In this example, the log record number is 27 (which corresponds to the value in LastProcessedSecLogRecordNumber - 0x1b).

Changing the Location of the Central Logs

The location of the central logs can be changed using the console and the Log Configuration tab. Specify the location in the Write Log File to Directory field.



Changing the Location of the Component Logs

The location of logs for Java components, that is, System Manager, Central Logger, and connectors, is changed by modifying the `com.sun.directory.wps.logging.directory` property in the `WatchList.properties` file.

In the following example, the logging directory for the Active Directory Connector is changed from `/var/opt/SUNWissw/logs/CNN101` to `/export/issw/logs/adConnector`:

```
Original WatchList.properties entry:
process.name[2]=CNN101
process.command[2]=/usr/java/bin/java
-Xmx256m -Xrs -DimqConnectionType=TLS
-Djava.library.path=/opt/SUNWissw/lib:/usr/lib/mps/secv1
-Djava.util.logging.config.file=/var/opt/SUNWissw
/resources/Log.properties
-Dcom.sun.directory.wps.logging.directory=/var/opt/SUNWissw/logs/CNN101
-DPSWHOME=/var/opt/SUNWissw -DWPCNFG=resources
-classpath /opt/SUNWissw/lib/common.jar:/opt/SUNWissw/lib/connector.jar:
/opt/SUNWissw/lib/db.jar:/opt/SUNWissw/lib/ldapjdk.jar:
/opt/SUNWissw/lib/manager.jar:/opt/SUNWissw/lib/registry.jar:
/opt/SUNWissw/lib/watchdog.jar:/var/opt/SUNWissw/resources:
/opt/SUNWissw/locale/resources:/usr/share/lib/jms.jar:
/usr/share/lib/imq.jar:/usr/share/lib/mps/secv1/jss3.jar:
/usr/sfw/share/lib/xerces-200.jar:.
com.sun.directory.wps.controller.AgentHarness CNN101
process.delay[2]=120000
process.interval[2]=60000
```

Note – The `idsync printstat` command provides information about the Connector ID and the installation location that can be used to locate the correct entry in the `WatchList.properties` file.

To change the logging directory, the command line entry for this connector must be edited as follows:

```
process.name[2]=CNN101
process.command[2]=/usr/java/bin/java -Xmx256m -Xrs
-DimqConnectionType=TLS
-Djava.library.path=/opt/SUNWisd/lib:/usr/lib/mps/secv1
-Djava.util.logging.config.file=/var/opt/SUNWisd/resources/Log.properties
-Dcom.sun.directory.wps.logging.directory=/export/isw/logs/adConnector
-DPSWHOME=/var/opt/SUNWisd -DWPSCNFG=resources
-classpath /opt/SUNWisd/lib/common.jar:/opt/SUNWisd/lib/connector.jar:
/opt/SUNWisd/lib/db.jar:/opt/SUNWisd/lib/ldapjdk.jar:
/opt/SUNWisd/lib/manager.jar:/opt/SUNWisd/lib/registry.jar:
/opt/SUNWisd/lib/watchdog.jar:/var/opt/SUNWisd/resources:
/opt/SUNWisd/locale/resources:/usr/share/lib/jms.jar:
/usr/share/lib/imq.jar:/usr/share/lib/mps/secv1/jss3.jar:
/usr/sfw/share/lib/xerces-200.jar:.
com.sun.directory.wps.controller.AgentHarness CNN101
process.delay[2]=120000
process.interval[2]=60000
```

After this change is made, the Identity Synchronization for Windows daemon (on Solaris) or service (on Windows) where the connector is running is stopped and restarted for the changes to take effect.

Isolating Problems in Directory Server

For detailed conceptual and configuration information related to logging options in Directory Server, see the following documents:

- <http://docs.sun.com/source/817-5221/logs.html>
- <http://docs.sun.com/source/817-5235/errcd.html>
- <http://docs.sun.com/source/817-5235/config.html#wp20751>

Isolating Problems in Message Queue

For information about understanding and configuring the Message Queue Broker log, see the *Message Queue Administration Guide* (<http://docs.sun.com/db/doc/817-3727>).

Isolating Problems in Active Directory

Active Directory logs information to the Directory Service event log on each domain controller. To increase the amount of logging, change the Windows registry entries. There are several diagnostic values under the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NTDS\Diagnostics` key. They default to 0 = no logging. Other permitted values are as follows:

- 1 = minimum
- 3 = medium
- 5 = maximum

Note – To isolate problems with Identity Synchronization for Windows, the values to set are Directory Access and LDAP Interface Events.

Glossary

accessor	A connector layer that interfaces directly with a directory source over protocols such as LDAP. Identity Synchronization for Windows has separate accessor implementations for Directory Server, Active Directory, and Windows NT. The accessor is often referenced in log messages about an action.
acknowledgement	A specialized message that acknowledges receipt of a message from another component. Identity Synchronization for Windows uses acknowledgements between connectors and Message Queue, and between the connector components (agent, controller, and accessor) to ensure that all changes are synchronized reliably.
action	An encapsulation of a single synchronization event. Identity Synchronization for Windows Connectors use actions to communicate user change events. Each action includes a type (such as CREATE, MODIFY, or DELETE) and enough attributes from the user entry to allow the destination connector to synchronize the change. All actions are processed atomically.
agent	A connector component that interfaces with Message Queue and translates attributes between their Directory Server names and Windows names. The agent is often referenced in log messages about an action.
attribute	Descriptive information about an entry that has a label and a value. Each attribute also follows a standard syntax for the type of information that can be stored as the attribute value.
attribute list	A list of required and optional attributes for a given entry type or object class.
audit log	<p>A central log file that contains entries for day-to-day events, such as a user's password being synchronized. Administrators can use the Identity Synchronization for Windows Console to control how many entries and what level of detail will be displayed in this log.</p> <p>Each connector produces an audit log of the users processed by that connector, and there is a centralized audit log that contains an aggregation of the audit logs produced by all of the connectors in the deployment.</p>
authentication	The process of proving the identity of the client user to Directory Server. A user must provide a bind DN and the corresponding password to be granted access to the directory. Directory

Server allows the user to perform functions or access files and directories based on the permissions granted to that user by the directory administrator.

authentication certificate

A digital file, issued by a third party, that cannot be transferred or forged. Authentication certificates are sent from server to client (or from client to server) to verify and authenticate the other party.

Auxiliary object class

An object class that augments the selected structural class, which provides additional attributes for synchronization. See [Structural object class](#).

base DN

Base distinguished name. A search operation is performed on the base DN, the DN of the entry, and all entries below it in the directory tree. For Active Directory and Directory Server, Synchronization User Lists are rooted at a specific base DN. All users under this base DN will be synchronized unless they are explicitly excluded by a filter.

bind DN

Bind distinguished name, The name used to authenticate to an LDAP directory (for example, Active Directory or Directory Server) when performing an operation.

Broker

See [Sun Java System Message Queue Broker](#).

CA

See [Certificate Authority \(CA\)](#).

cascading replication

In a cascading replication scenario, one server (often called the *hub supplier*) acts both as a consumer and a supplier for a particular replica. The server holds a read-only replica and maintains a change log. Hub supplier receives updates from the supplier server that holds the master copy of the data, and in turn supplies those updates to the consumer.

Central Logger

A Core component that manages all of the central logs, which are an aggregation of every connector's audit and error logs. Administrators can monitor the health of an entire Identity Synchronization for Windows installation by monitoring these logs, and can view them directly or from the Identity Synchronization for Windows Console. By default, the central logs are available on the machine where Core was installed in the `<install-root>\logs\central\` subdirectory.

certificate

A collection of data that associates public keys with a network identity. This information enables the recipient of an electronic message to verify the authenticity of the message and the message sender. When configuring Identity Synchronization for Windows Connectors to use SSL communication, an administrator must add certificates to the Connector certificate databases before trusted SSL communication can occur. See also [Certificate Authority \(CA\)](#).

Certificate Authority (CA)

A company or organization that sells and issues authentication certificates. An administrator may purchase an authentication certificate from a Certificate Authority that the administrator trusts. A root Certificate Authority certificate is used to sign other certificates. When

configuring an Identity Synchronization for Windows Connector to use SSL, the administrator must add the appropriate root Certificate Authority certificate to the Connector's certificate database.

certificate database	A secure repository for certificates, which includes three files: <code>cert8.db</code> , <code>key3.db</code> , and <code>secmod.db</code> . In Identity Synchronization for Windows, each connector has its own certificate database directory (for example, <code><install-root>/etc/CNN100</code>). See also certificate .
character type	Character type distinguishes alphabetic characters from numeric (or other) characters and the mapping of uppercase to lowercase letters.
client	See LDAP client .
command-line interface (CLI)	A means of communication between a program and its user, based solely on textual input and output. Commands are input using a keyboard or similar device, and are interpreted and executed by the program. The Identity Synchronization for Windows command-line interface is named <code>idsync</code> and is available in the <code>bin/</code> directory where Core is installed.
configuration directory	A special installation of Directory Server that serves as a repository for configuration and status information. Identity Synchronization for Windows stores all of its configuration within the configuration directory chosen during Core installation.
configuration password	A password chosen during Core installation that protects all sensitive Identity Synchronization for Windows information stored in the configuration directory. The configuration password must be provided when using the installer, the console, or the command-line interface.
configuration registry	Another term used by Identity Synchronization for Windows to refer to the configuration directory .
connector	A Java process that manages the interaction of Identity Synchronization for Windows with a single data source (such as a Directory Server, an Active Directory domain, or a Windows NT domain). A connector detects user changes in the data source and publishes these changes to remote connectors over Message Queue. A connector also subscribes to user change topics and applies updates from these topics to the data source.
console	A Graphical User Interface used to configure and monitor server applications. Sun Java System Directory Server and Identity Synchronization for Windows have separate consoles.
controller	A connector component that interfaces with the agent and accessor components. The controller performs key synchronization-related tasks such as determining a user's membership in a Synchronization User List, searching for and linking equivalent user entries, and detecting changes to users by comparing current user entries with the previous versions stored in the object cache. The controller is often referenced in log messages about an action.

Core	The first Identity Synchronization for Windows component that is installed. The Core includes the initial configuration stored in the configuration directory, the System Manager, the Central Logger, the Console, and the command-line interface.
creation attributes	Attributes that are synchronized only when an object is created. All significant attributes are automatically synchronized when an object is created. An administrator can configure default values for creation attributes that might not have a corresponding attribute value in the remote directory.
daemon	A background process on a UNIX system that is responsible for a particular system task. Daemon processes do not need human intervention to continue functioning. Connectors, the System Manager, and the Central Logger run as daemon processes that are launched and monitored by the Identity Synchronization for Windows Watchdog.
directory information tree (DIT)	The logical representation of the information stored in the directory that mirrors the tree model used by most file systems, where the tree's root appears at the top of the hierarchy.
Directory Manager (DM)	The privileged directory server administrator, comparable to the root user on a UNIX system. Identity Synchronization for Windows requires Directory Manager credentials to perform certain configuration operations, but the connector does not require Directory Manager credentials for synchronization.
directory source	A Sun Java System Directory Server, Windows Active Directory domain, or Windows NT domain. Directory sources contain users to be synchronized.
DIT	See directory information tree (DIT) .
DM	See Directory Manager (DM) .
DNS	See Domain Name System (DNS) .
domain	<p>(1) (n.) The last part of a fully qualified domain name that identifies the company or type of organization that owns the domain name, for example, <code>company.com</code>, <code>host.example.edu</code>.</p> <p>(2) (n.) Resources under control of a single computer system.</p>
domain controller	A Windows server that stores user account information, authenticates users, and enforces security policy for a Windows domain. Identity Synchronization for Windows Connectors communicate directly with domain controllers to detect changes to user accounts and to synchronize changes made in Directory Server user entries.
Domain Name System (DNS)	System used by machines on a network to associate standard IP addresses (such as 198.68.0.0) with host names (such as <code>www.example.com</code>). Machines normally get the IP address for a host name from a DNS server or look up the address in tables maintained on their systems.

file extension	The portion of a file name following the dot (.) that typically defines the file type . For example, in a file named <code>index.html</code> , the file extension is <code>html</code> .
file type	The format of a given file. For example, graphics files are often saved in GIF format, while a text file is usually saved as in ASCII text format. File types are usually identified by the file extension , for example, <code>.text</code> .
FSMO role	Flexible Single-Master Operation role. Mechanism used by Active Directory to prevent update conflicts in multimaster deployments. Some objects are updated in a single-master mode even if the deployment is multimaster, which is very similar to the old concept of a Primary Domain Controller (PDC) in Windows NT domains. There are five FSMO roles in an Active Directory deployment, but only the PDC emulator role affects Identity Synchronization for Windows. Because password updates are replicated immediately only to the Active Directory domain controller with the PDC emulator role, Identity Synchronization for Windows uses this domain controller for synchronization. Otherwise, synchronization with Sun Java System Directory Server might be delayed for several minutes.
global catalog	A Windows repository that stores Active Directory topology and schema information for Active Directory directories.
host name	A name for a machine in the form <i>machine.domain.com</i> , which is translated into an IP address. For example, <code>sales.example.com</code> is the machine <code>sales</code> in the subdomain <code>example</code> , and domain <code>com</code> .
Identity Synchronization for Windows Console	A graphical user interface used to configure and monitor Identity Synchronization for Windows.
inbound	Within the connector, the direction of actions that flow from a directory source toward Message Queue. Changes detected by the connector flow inbound into the system. Log messages about an action often refer to events that occur on the inbound side of the connector.
IP address	Internet Protocol address. A set of numbers, separated by dots, that specifies the actual location of a machine on the Internet (for example, <code>192.168.0.0</code>).
ISO	International Standards Organization.
LDAP	See Lightweight Directory Access Protocol (LDAP) .
LDAP client	Software used to request and view LDAP entries from an LDAP Directory Server. Identity Synchronization for Windows Connectors act as LDAP clients when connecting to LDAP servers.
LDAP URL	A URL that provides the means of locating directory servers using DNS and then completing the query using LDAP. A sample LDAP URL is <code>ldap://ldap.example.com</code> .

Lightweight Directory Access Protocol (LDAP)	The directory service protocol designed to run over TCP/IP and across multiple platforms. Identity Synchronization for Windows uses LDAP to communicate with Active Directory domain controllers and Sun Java System Directory Servers.
locale	The locale identifies the collation order, character type, monetary format, and time and date format used to present data for users of a specific region, culture, or custom. This includes information about how data of a given language is interpreted, stored, or collated. The locale also indicates which code page should be used to represent a given language.
main object class	See Structural object class .
Message Queue	See Sun Java System Message Queue .
MMR	See multimaster replication (MMR) .
MQ	See Sun Java System Message Queue .
multimaster replication (MMR)	A Directory Server replication model in which entries can be written and updated on any of several master replica copies without requiring communication with other master replicas before the write or update is performed. Modifications made on one server are automatically replicated to the other servers. Identity Synchronization for Windows can be installed in a deployment with multiple preferred Directory Servers. However, when synchronizing changes to Windows, the preferred Directory Server must be available, and when synchronizing changes from Windows, the preferred or secondary Directory Server must be available.
naming context	A specific suffix of a directory information tree (DIT) that is identified by its distinguished name (DN), for example, dc=example, dc=com. In Identity Synchronization for Windows, a directory source for Sun Java System Directory Server is defined by the suffix containing the data to be synchronized. Also known as root suffix.
object cache	An in-process database used by the Windows Connectors to detect changes to user entries. The object cache stores a hashed summary of each user entry, which enables Windows Connectors to determine which specific attributes in the user entry have changed.
object class	A template specifying the kind of object that the entry describes and the set of valid and mandatory attributes that the entry contains. For example, Directory Server specifies an inetorgperson object class that has attributes such as cn and userpassword.
on-demand password synchronization	A mechanism whereby a user's password in Directory Server is not updated until the user attempts to authenticate to Directory Server. The user's password is synchronized only if the provided password matches what is stored in Active Directory.

outbound	Within the connector, the direction of actions that flow from Message Queue toward the directory source. Changes applied by a connector flow outbound into the synchronized directory source. Log messages about an action often refer to events that occur on the outbound side of the connector.
password file	A file on UNIX systems that stores UNIX user login names, passwords, and user ID numbers. Also known as <code>/etc/passwd</code> because of its location.
password policy	A set of rules that govern how passwords are used in a given directory.
permission	In the context of access control, the permission states whether access to the directory information is granted or denied, and the level of access that is granted or denied.
plug-in	<p>An accessory program that can be loaded and then used as part of the overall system.</p> <p>For example, Identity Synchronization for Windows uses the Directory Server Plug-in to enhance Directory Server Connector change-detection features and to provide bidirectional support for password synchronization between Active Directory and Directory Server.</p>
preferred Directory Server	A main Directory Server instance used by Identity Synchronization for Windows to detect and apply changes to user entries. While this server is available, Identity Synchronization for Windows will not communicate with any other Directory Servers.
protocol	A set of rules that describes how devices on a network exchange information.
RCL	See retro changelog.
resync interval	How often a connector checks a directory source for changes. This periodic check is efficient and only requires reading entries of users that have changed since the last check. The console expresses this value in milliseconds and provides 1000 (1 second) as a default.
retro changelog (RCL)	A Directory Server database (<code>cn=changelog</code>) that stores a record of all changes made to Directory Server. Identity Synchronization for Windows uses the retro changelog to detect changes made to Directory Server. In an MMR environment, the retro changelog must be enabled on the preferred Directory Server.
root	The most privileged user available on UNIX systems (also called superuser). The root user has complete access privileges to all files on the system. On Solaris systems, Identity Synchronization for Windows must be installed as root.
root suffix	The parent of one or more LDAP sub-suffixes. A directory tree can contain more than one root suffix.

schema	Definitions describing what types of information can be stored as entries in a directory. When information that does not match the schema is stored in the directory, clients attempting to access the directory might be unable to display the proper results.
schema checking	Ensures that entries added or modified in the directory conform to the defined schema. Schema checking is on by default, and users will receive an error if they try to save an entry that does not conform to the schema.
secondary Directory Server	A Directory Server instance in an MMR environment that Identity Synchronization for Windows can use when the preferred Directory Server is not available. While the preferred Directory Server is unavailable, Identity Synchronization for Windows can synchronize changes made in Active Directory or Windows NT to the secondary Directory Server, but changes made at the secondary Directory Server or any other Directory Server will not be synchronized until the preferred Directory Server is available.
Secure Sockets Layer (SSL)	A software library used for establishing a secure connection between two parties (client and server). Used to implement HTTPS (the secure version of HTTP) and LDAPS (the secure version of LDAP).
Server Console	Java-based application that allows you to perform administrative management of your Directory Server from a GUI.
server root	A directory on the server machine dedicated to holding the server program configuration, maintenance, and information files.
service	A background process on a Windows machine that is responsible for a particular system task. Service processes do not need human intervention to continue functioning. On Windows, connectors, the System Manager, and the Central Logger run as processes that are launched and monitored by the Identity Synchronization for Windows Watchdog service.
significant attributes	Attributes that are synchronized when an entry is created or modified.
SSL	Secure Sockets Layer (SSL) .
Structural object class	The primary object class of an entry that defines the set of valid and mandatory attributes on the user entries that Identity Synchronization for Windows synchronizes. For example, the default Active Directory object class is <code>user</code> , and the default Directory Server object class is <code>inetorgperson</code> . See also Auxiliary object class
subcomponent	A lightweight process or library that runs separate from a connector. A subcomponent runs close to the directory source that a connector manages, and enables functionality in the connector that cannot be achieved on a remote machine or separate process. The subcomponent communicates with the connector over a custom encryption channel to receive configuration information, report change events, and log to the Central Logger. Identity

	Synchronization for Windows includes three subcomponents: the Directory Server Plug-in, the Windows NT Password Filter DLL, and the Windows NT Change Detector.
suffix	The name of the entry at the top of the directory tree, below which data is stored. Multiple suffixes are possible within the same directory. Each database has only one suffix.
SUL	See Synchronization User List (SUL) .
Sun Java System Message Queue	An enterprise messaging system that implements the Java Message Service (JMS) open standard. The basic architecture of Message Queue consists of publishers and subscribers that exchange messages by way of a common service. Sun Java System Message Queue is administered by a dedicated message broker, which is responsible for controlling access to Message Queue, maintaining information about active publishers and subscribers, and ensuring that messages are delivered. Identity Synchronization for Windows uses Message Queue to securely synchronize user change events, distribute configuration information, and monitor the health of remote components.
Sun Java System Message Queue Broker	A stand-alone Java server that provides clients access to Sun Java System Message Queue. On Solaris systems, the Broker is controlled via the <code>/etc/init.d/imq</code> daemon script, and on Windows systems, it is controlled with the “iMQ Broker” service. Identity Synchronization for Windows configures and starts the Broker during Core installation.
superuser	See root .
synchronization host	Server that stores synchronized data according to the rules defined in the Synchronization User Lists.
Synchronization User List (SUL)	A list that defines users in the Sun and Windows directories to be synchronized. An SUL can restrict the scope of users to be synchronized based on an LDAP base DN or filter.
synchronized attributes	See significant attributes .
System Manager	A stand-alone Java process that is started by the Watchdog daemon (on Solaris) or service (on Windows) where Core is installed. The System Manager distributes configuration information to the connectors and Central Logger, monitors the health of the system, and coordinates idsync resync operations.
topology	The way a directory tree is divided among physical servers and how these servers link with one another.
uid	A unique number associated with each user on a UNIX system.
URL	Uniform Resource Locator. The addressing system used by the server and the client to request documents. It is often called a location. The format of a URL is

[protocol]://[machine:port]/[document]. The port number is necessary only on selected servers, and it is often assigned by the server, freeing the user of having to place it in the URL.

Watchdog

A stand-alone Java process that is installed on every machine where Core or a connector is installed. The Watchdog starts all Identity Synchronization for Windows Java processes including the System Manager, the Central Logger, and Connectors. If any of these components fail, the Watchdog restarts them. On Solaris systems, the Watchdog is controlled using the `/etc/init.d/isw` daemon script. On Windows systems, it is controlled with the “Sun Java System Identity Synchronization for Windows” service.

Index

Numbers and Symbols

%TEMP% environment variable, 162

A

accessor connector layers, 152

accessor.level, 160

accessor.saint.level, 160

accounts, new user, 27

action ID types, 152

actions, 152, 155, 157, 169, 173, 175

Active Directory

- changing passwords, 144

- connector layers, 152

- deleted entries, 27

- deployment considerations, 20

- domains, 25

- failover configuration, 84

- global setting, 85

- inactive accounts, 27

- new users, 27

- password changes, 144

- samaccountname, 25

- special users, 29

- synchronization user lists, 43-44

activedirectorydomainname attribute, 60

administration rights, 29-30

agent connector layers, 152

agent.level, 160

agentout.level, 161

attributes

- activedirectorydomainname, 20, 60

- configuring, 36-37

- data, 152

- destinationindicator, 52, 56, 57, 59, 60

- dspswuserlink, 27, 57, 62

- dspswvalidate, 57

- mapping, 38-39

- mappings, 38-39

- modification settings, 37-38

- nsslapd-infolog-area, 157

- nsslapd-infolog-level, 158

- other, 152

- pwdlastset, 155

- samaccountname, 25, 54, 71

- shadowmax, 42

- shadowmin, 42

- shadowwarning, 42

- uid=PSWConnector, 29

- user_name, 25

- user_nt_domain_name, 60

- userPassword, 58

audit.log, 154

audit logging, 151

authentication

- Active Directory, 117

- denying, 128

- domain controllers, 112

- editing /etc/nsswitch.conf, 128

- editing /etc/pam.conf, 128, 129

- for failover, 112

- on Solaris, 24, 126

authentication (*Continued*)

- on Windows, 122
 - over TLS, 137
 - rules, 128
 - using PAM, 126, 128
- authentication certificates, 170

B

- Base DNs, 120
- brokers, Message Queue, 100, 101, 103, 177

C

Central Logger

- enabling debug logging, 158
 - log locations, 165
 - out-of-order messages, 155
 - plug-in logging, 114
 - processing logs, 161
- central logs, 157, 165, 169
-
- centralLogger component, 161
-
- Certificate Authority, 21, 91, 170
-
- certificate databases, 21, 91, 96, 171
-
- certificates
- Active Directory, 91
 - Active Directory SSL, 21
 - adding to certificate databases, 96
 - authentication, 170
 - Certificate Authority, 21, 91
 - configuring connectors, 137
 - definition, 170
 - Directory Server, 91, 96
 - trusted, 73, 82, 91, 96
 - using Certificate Authorities, 91
 - using with plug-ins, 82
 - using with SSL, 21, 73, 82, 91
- change detection
- action paths, 153, 173, 175
 - Active Directory, 61, 103, 108, 146
 - connectors, 20, 30, 76, 108, 109, 110, 112, 171
 - controller, 153, 171
 - Directory Server, 20, 30, 61, 76, 110, 175

change detection (*Continued*)

- failover, 112
 - failures, 30, 112
 - Identity Manager, 144, 145
 - Identity Synchronization for Windows, 109, 143, 144, 152
 - pwsync command, 144, 145
 - SUL filters, 43, 49
 - Windows NT, 162
 - Windows NT Change Detector, 151, 162
- Change Detector, Windows NT, 151
-
- commands
- idsync certinfo, 91, 95
 - idsync printstat, 87, 90, 111, 159
 - idsync resync, 20, 52, 66, 73, 80, 97, 100, 112, 149, 157
 - idsync startsync, 88, 91, 114
 - idsync stopsync, 114
- common.level, 161
-
- components
- changing logs location, 165-166
 - Core, 170, 177, 178
 - logging, 161
 - trusting certificates, 96
- configuring
- attribute mappings, 38-39
 - Directory Server, 30, 70
 - failover domain, 34
 - Identity Manager, 109
 - preferred Directory Server, 88
 - shadowAccount object class, 41
 - synchronization user lists, 43-44
 - Windows NT, 35
- connectors
- definition, 171
 - initializing connector state, 110, 111
 - installing, 74, 100
 - layers, 152
 - modification rate, 20
 - persisting states, 111
 - security requirements, 21, 73, 137
 - uninstalling, 100
 - updating, 108
 - using in large deployments, 80

- connectors (*Continued*)
 - WAN deployments, 60
- console.log, 162
- consoles
 - debug logging, 162
 - validations, 49
- controller connector layers, 152
- controller.level, 160
- Core
 - components, 170, 177, 178
 - configuring Directory Server, 81
- CREATE actions, 152
- creating users, 26, 29
- credentials
 - Active Directory, 113
 - bind, 133, 137
 - candidate, 137
 - Directory Manager, 172
 - problems, 100
 - proxy, 128
 - specifying, 33, 128
- crom.level, 161

- D**
- daemon, Identity Synchronization for Windows, 98, 99, 114, 158, 159, 172
- data attributes, 152
- databases
 - certificate, 21, 91, 96, 171
 - object cache, 103, 108, 110, 112, 155, 174
 - retro changelog, 175
 - suffixes, 177
- debug.log, 159, 160
- debug logging, 100, 151
- debug page, 146, 147
- debugErrors.log, 160
- decryption routines, 161
- DELETE actions, 152
- deleting users, 26
- deployment
 - examples, 23, 69-74
 - HA environments, 74
 - increasing connector worker threads, 97
- deployment (*Continued*)
 - large deployments, 80
 - performance requirements, 20
 - preparing for, 20
 - primary/failover configurations, 98
 - running idsync resync, 52, 100
 - synchronization direction, 20
 - with Identity Manager, 141
- destinationindicator attribute, 52, 56, 57, 59, 60
- directory resources, deployment example, 23
- Directory Server
 - changing passwords, 144-145
 - configuring, 30, 70
 - configuring LDAP repository for PAM, 119
 - connector layers, 152
 - deleted entries, 27
 - deployment considerations, 20
 - deployment example, 24
 - hierarchical DIT, 28
 - idsync resync, 49
 - in HA environment over WAN, 69
 - in MMR environments, 23
 - inactive accounts, 27
 - isolating problems, 166
 - LINK actions, 152
 - masters, 20
 - new users, 27
 - PAM, 24
 - populating LDAP repository for PAM, 123
 - preferred source, 31
 - preventing password change propagation, 146-147
 - product documentation links, 15
 - purpose, 24
 - secondary server, 31
 - uid, 25
 - using with Identity Manager, 145
 - verifying synchronization, 132
- Directory Server Plug-in, 20, 114, 151, 157
- directory sources, specifying, 31
- disabled accounts, 27
- distinguished names, 170
- DNS, 119, 120, 127, 128
- documentation
 - audience, 13

documentation (*Continued*)

- overview, 14-15
 - recommended reading, 14
- domain controllers, 147
- architecture example, 23
 - authenticating, 112
 - definition, 172
 - deployment example, 70
 - failover, 34
 - for WAN deployment, 60
 - logs, 167
 - PDC, 138, 142
 - PDC FSMO role owners, 34, 115
 - uSNChanged values, 102
 - with SSL, 91, 95
- dspswuser auxiliary object class, 57
- dspswuserlink attribute, 27, 57, 62
- dspswvalidate attribute, 57

E

- encryption keys, 98, 114, 115
- encryption.level, 161
- environment variables
 - %TEMP%, 162
 - ISW_DEBUG_INSTALL, 161
- environments
 - Active Directory, 117
 - changing passwords, 72
 - high-availability over WAN, 69-116
 - Identity Synchronization for Windows and Identity Manager cohabitation, 141
 - MMR, 23, 175, 176
 - NT, 117
 - using PAM, 117
 - WAN, 61
 - Windows NT, 138
- Example Bank deployment example, 23
- executing, periodic resynchronization, 104-109

F

- failover
- Active Directory connector, 110
 - Directory Server connector, 110
 - domain controllers, 34
 - installation, 88, 99, 108, 109, 112, 114, 116
 - maintenance, 112
 - monitoring logs, 115-116
 - starting synchronization, 114
 - stopping synchronization, 113
- failures, detecting, 109
- features, 19, 28
- DIT, 28
 - moving users, 28
 - SUL filters, 28
 - WAN deployments, 28
- filters, using, 43, 49, 80, 101
- form property, configuring, 145-146

G

- Global Catalog, 32, 173
- global setting, Active Directory, 85
- Global Telco
- configuration, 74
 - configuring Directory Server, 81
 - on-demand synchronization, 84
 - primary installation, 74
 - requirements, 72
- Globally Unique Identifier. *See* GUIDs, 27
- GUIDs, 26, 27, 152

H

- High Encryption Pack, 20
- hub replicas, 20, 70

I

- Identity Manager
- cohabitation with Identity Synchronization for Windows, 141-149

Identity Manager (*Continued*)

- Global Telco case study, 70, 104
- password changes, 73, 145
- Identity Synchronization for Windows
 - failover installation, 88, 99, 109, 114, 116
 - primary installation, 74, 81, 100, 101, 104, 109, 113, 116
- idsconfig, 120, 121, 128, 137
- idsync certinfo command, 91, 95
- idsync command line interface, 171
- idsync printstat command, 87, 90, 111, 159
- idsync resync command
 - central log file, 157
 - deployment example, 52
 - failover maintenance, 112
 - increasing connector worker threads, 97
 - initial operation, 100
 - linking users, 20, 49, 66, 73, 80, 149
 - primary installation, 104
 - reducing peak load, 80
- idsync startsync command, 88, 91, 114
- idsync stopsync command, 114
- inactive accounts, 27
- increasing connector work threads, 97
- initializing connector state, 110
- installation
 - connectors, 74
 - failover, 88
- ISW_DEBUG_INSTALL environment variable, 161

L

- LANs, 60
- layers, connector, 152
- LDAP
 - accessor, 152
 - attributes, 42
 - communicating with domain controllers, 75
 - deployment considerations, 20
 - filters, 80, 101
 - maximum connections, 98
 - PAM, 24, 26, 41, 117
 - sample URL, 173
 - shadowAccount object class, 41

- LINK actions, 152
- linking users
 - idsync resync, 52
 - new users, 80
- log.properties level, 161
- logging
 - audit and debug, 151
 - central logs, 165
 - centralLogger, 161
 - component logs, 165
 - components, 161
 - configuring Directory Server plug-in, 157
 - console logs, 162
 - debug, 100
 - levels, 100
 - message samples, 154
 - monitoring logs, 115-116
 - security logs, 164
 - types, 161
- login IDs, 26

M

- maintenance, failover, 112
- manager.level, 161
- Message Queue
 - agent, 153
 - broker description, 177
 - broker memory limits, 100, 101, 103
 - isolating problems, 167
- Microsoft
 - Knowledge Base Articles, 15
 - publications, 15
- migrating
 - Example Bank case study, 25
 - from NT to Active Directory, 58, 61
 - user passwords, 65
- MMR environments, 23, 74, 175, 176
- modification rate, 20
- MODIFY actions, 152
- monitoring logs, 115-116
- mq.level, 161
- Multi-Master Replication. *See* MMR environments, 23
- multiple object classes, 28

multiple passwords, setting, 99

N

new users

- Active Directory, 27
- creating, 20
- Directory Server, 27

O

object cache databases, 103, 108, 110, 112, 155, 174

object classes

- dspswuser, 57
- shadowAccount, 41, 60

on-demand synchronization, 84

P

PAM

- configuring LDAP repository, 119
- Directory Server, 24
- populating LDAP repository, 123
- verifying synchronization, 132

Password Filter DLL, 151, 162

passwords

- changing on Active Directory, 144
- changing on Directory Server, 144-145
- changing through Identity Manager, 145
- dspswvalidate attribute, 57
- encrypting, 76, 98
- Example Bank, 25
- losing changes, 99
- managing on Solaris, 24
- on-demand synchronization, 34, 76, 90, 95, 108, 112
- preventing display in command line, 104
- preventing propagation, 146-147
- propagating changes, 73
- replicating to PDC FSMO role owner, 34
- resetting, 104
- running idsync resync, 100
- setting multiple, 99

passwords (*Continued*)

- shadowmax attribute, 42
- shadowmin attribute, 42
- shadowwarning attribute, 42
- specifying, 32
- synchronizing, 25, 52, 73
- userPassword attribute, 57

PDC, 24, 138, 142

PDC FSMO role owners, 34, 60, 71, 83, 90, 115

peak modification rate, 20

periodic resynchronization, 104-109, 108-109

permissions, 29, 104, 170, 175

plug-ins

- Directory Server, 20, 114-115, 151, 157
- Retro Changelog, 20

Pluggable Authentication Module. *See* PAM, 24

preparing for deployment, 20

Primary Domain Controller. *See* PDC, 138

primary installation

- configuring Directory Server, 81
- Identity Synchronization for Windows, 74, 81, 100, 101, 104, 109, 113, 116

provisioned users, 148-149

psloglist.exe, 164

publications

- Microsoft, 15
- related, 14

pwdlastset attribute, 155

pwsync, configuring, 146-147

R

read-only replicas, 20, 70

recommended reading, 14

REFRESH actions, 152

reinstalling Directory Server plug-ins, 114-115

related documentation, 14

replicas

- hub, 20, 70
- read-only, 20, 70

replication agreements, 70

requirements

- Example Bank, 25
- Global Telco, 72

resync-batch.pl script, 102
 resync.log, 157
 resyncDebug.log, 159, 160
 resynchronization, 100-103
 retro changelog, 110, 175
 Retro Changelog Plug-in, 20
 rights, administration, 29
 role owner, PDC FSMO, 115

S

s requirements, 25
 samaccountname attribute, 54, 71
 secondary servers, 31
 Secure Sockets Layer. *See* SSL, 15
 security
 configuring, 82, 137
 considerations, 21, 160
 logs, 164
 using TLS, 118
 SENTINEL actions, 152
 setting up
 Identity Manager, 147-148
 SSL, 91-96
 Shadow Account
 shadowmax, 42
 shadowmin, 42
 shadowwarning, 42
 shadowAccount auxiliary object class, 41, 60
 special users
 administration rights, 29-30
 creating, 29
 delegating control, 29
 permissions, 29
 specifying
 credentials, 33
 directory sources, 31
 passwords, 32
 secondary servers, 31
 SSL
 certificates, 21
 enabling, 20, 21
 requirements, 21, 73
 setting up, 91

SSL (*Continued*)

 with trusted certificates, 21, 73, 82, 91
 starting synchronization failover, 114
 suffixes, 177

SULs

 description, 177
 determining membership, 153
 purpose, 26
 using filters, 43, 49
 validating filters, 49

Sun Java™, 13, 15

synchronization

 starting, 114
 stopping, 113
 synchronization settings, 37-38
 attribute mapping, 38-39
 attribute settings, 36-37
 shadowAccount object class, 41

Synchronization User Lists. *See* SUL, 26

synchronizing

 inactive accounts, 27
 passwords, 25
 users, 26

SYNCING state, 111

System Directory Server. *See* Directory Server, 15

System Identity Synchronization for Windows. *See* Identity Synchronization for Windows, 13

System Manager

 checking status, 88, 91, 111
 component logs location, 165
 description, 177
 enabling debug logging, 158
 new configurations, 98
 processing logs, 161

System Message Queue. *See* Message Queue, 15

T

tasks.level, 161

TLS, 118, 137

Transport Layer Security. *See* TLS, 118

troubleshooting, 151

trust requirements, 94

trusted certificates, 21, 73, 82, 91, 96

U

- UNKNOWN actions, 152
- unlinking migrated users, 62
- user accounts, eliminating, 26
- user_nt_domain_name attribute, 60
- user passwords
 - Change My Password Form, 146
 - Change Password Form, 146
 - Change User Password Form, 146
 - changing on Active Directory, 144
 - changing on Directory Server, 144-145
 - changing through Identity Manager, 145
 - dspswvalidate attribute, 57
 - encrypting, 76, 98
 - Example Bank, 25
 - losing changes, 99
 - migrating, 65
 - modifying, 111
 - on-demand synchronization, 34, 76, 90, 95, 108, 112
 - preventing display in command line, 104
 - preventing propagation, 146-147
 - propagating changes, 73, 144, 145
 - replicating to PDC FSMO role owner, 34
 - Reset User Password Form, 146
 - resetting, 104
 - running idsync resync, 100
 - setting multiple, 99
 - shadowmax attribute, 42
 - shadowmin attribute, 42
 - shadowwarning attribute, 42
 - synchronizing changes, 36, 52, 73
 - userPassword attribute, 57, 59
- userPassword attribute, 58
- users
 - creating, 26, 29
 - deleting, 26
 - linking migrated, 65
 - moving between domains, 66
 - provisioned, 148-149
 - synchronizing, 26
 - unlinking migrated, 62
- uSNChanged values, 102

W

- WAN deployments, 60, 61, 74
- websites
 - Directory Server publications, 15
 - Identity Synchronization for Windows publications, 14
 - Message Queue publications, 15
- Windows
 - Global Catalog, 32, 173
- Windows NT
 - Change Detector, 151
 - configuring, 35
 - connector layers, 152
 - Password Filter DLL, 151
 - SUL_NT, 43-44
 - synchronization user lists, 43-44
 - user_name, 25

X

- xml
 - form properties, 146
 - link files, 54
 - linkusers-ad-only.cfg, 55, 56
- xml.level, 161