



Sun Java™ System

# Identity Synchronization for Windows 1 Deployment Planning Guide

---

2004Q3

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

Part No: 817-6200

Copyright © 2005 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

Sun Microsystems, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents> and one or more additional patents or pending patent applications in the U.S. and in other countries.

THIS PRODUCT CONTAINS CONFIDENTIAL INFORMATION AND TRADE SECRETS OF SUN MICROSYSTEMS, INC. USE, DISCLOSURE OR REPRODUCTION IS PROHIBITED WITHOUT THE PRIOR EXPRESS WRITTEN PERMISSION OF SUN MICROSYSTEMS, INC.

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.

This distribution may include materials developed by third parties.

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 in other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Java, Solaris, JDK, Java Naming and Directory Interface, JavaMail, JavaHelp, J2SE, iPlanet, the Duke logo, the Java Coffee Cup logo, the Solaris logo, the SunTone Certified logo and the Sun ONE logo 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 architecture developed by Sun Microsystems, Inc.

Legato and the Legato logo are registered trademarks, and Legato NetWorker, are trademarks or registered trademarks of Legato Systems, Inc. The Netscape Communications Corp logo is a trademark or registered trademark of Netscape Communications Corporation.

The OPEN LOOK and Sun(TM) 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.

Products covered by and information contained in this service manual are controlled by U.S. Export Control laws and may be subject to the export or import laws in other countries. Nuclear, missile, chemical biological weapons or nuclear maritime end uses or end users, whether direct or indirect, are strictly prohibited. Export or reexport to countries subject to U.S. embargo or to entities identified on U.S. export exclusion lists, including, but not limited to, the denied persons and specially designated nationals lists is strictly prohibited.

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 © 2005 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, Etats-Unis. Tous droits réservés.

Sun Microsystems, Inc. détient les droits de propriété intellectuels relatifs à la technologie incorporée dans le produit qui est décrit dans ce document. En particulier, et ce sans limitation, ces droits de propriété intellectuelle peuvent inclure un ou plusieurs des brevets américains listés à l'adresse <http://www.sun.com/patents> et un ou des brevets supplémentaires ou des applications de brevet en attente aux Etats - Unis et dans les autres pays.

CE PRODUIT CONTIENT DES INFORMATIONS CONFIDENTIELLES ET DES SECRETS COMMERCIAUX DE SUN MICROSYSTEMS, INC. SON UTILISATION, SA DIVULGATION ET SA REPRODUCTION SONT INTERDITES SANS L'AUTORISATION EXPRESSE, ECRITE ET PREALABLE DE SUN MICROSYSTEMS, INC.

Cette distribution peut comprendre des composants développés par des tierces parties.

Des parties de ce produit peuvent être dérivées des systèmes 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, Java, Solaris, JDK, Java Naming and Directory Interface, JavaMail, JavaHelp, J2SE, iPlanet, le logo Duke, le logo Java Coffee Cup, le logo Solaris, le logo SunTone Certified et le logo Sun[tm] ONE 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.

Legato, le logo Legato, et Legato NetWorker sont des marques de fabrique ou des marques déposées de Legato Systems, Inc. Le logo Netscape Communications Corp est une marque de fabrique ou une marque déposée de Netscape Communications Corporation.

L'interface d'utilisation graphique OPEN LOOK et Sun(TM) 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.

Les produits qui font l'objet de ce manuel d'entretien et les informations qu'il contient sont régis par la législation américaine en matière de contrôle des exportations et peuvent être soumis au droit d'autres pays dans le domaine des exportations et importations. Les utilisations finales, ou utilisateurs finaux, pour des armes nucléaires, des missiles, des armes biologiques et chimiques ou du nucléaire maritime, directement ou indirectement, sont strictement interdites. Les exportations ou réexportations vers des pays sous embargo des Etats-Unis, ou vers des entités figurant sur les listes d'exclusion d'exportation américaines, y compris, mais de manière non exclusive, la liste de personnes qui font objet d'un ordre de ne pas participer, d'une façon directe ou indirecte, aux exportations des produits ou des services qui sont régis par la législation américaine en matière de contrôle des exportations et la liste de ressortissants spécifiquement désignés, sont rigoureusement interdites.

LA DOCUMENTATION EST FOURNIE "EN L'ETAT" ET TOUTES AUTRES CONDITIONS, DECLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES, DANS LA MESURE AUTORISEE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE A LA QUALITE MARCHANDE, A L'APTITUDE A UNE UTILISATION PARTICULIERE OU A L'ABSENCE DE CONTREFACON.

# Contents

<b>List of Figures</b> .....	<b>7</b>
<b>List of Tables</b> .....	<b>11</b>
<b>Preface</b> .....	<b>13</b>
Who Should Use This Book .....	13
Before You Read this Book .....	14
Conventions Used in this Book .....	14
Typographic Conventions .....	14
Symbols .....	15
Shell Prompts .....	16
Related Documentation .....	16
Books in This Documentation Set .....	16
Other Documentation .....	17
Accessing Sun Resources Online .....	17
Contacting Sun Technical Support .....	18
Related Third-Party Web Site References .....	18
Sun Welcomes Your Comments .....	18
 <b>Chapter 1 Introduction</b> .....	 <b>19</b>
Deployment Considerations .....	19
 <b>Chapter 2 Case Study: Deploying in an Multi-Master Replication Environment</b> .....	 <b>23</b>
Overview .....	23
Existing Example Bank's Architecture .....	24
Directory Server Information .....	24
Windows NT Information .....	25
Active Directory Information .....	25
Understanding Example Bank's Technical Requirements .....	26
Identity Synchronization for Windows Features Discussed in this Case Study .....	28

Deploying the Solution .....	28
Creating a Special Active Directory User for Identity Synchronization for Windows .....	29
Configuring the Identity Synchronization for Windows Core .....	33
Configuring Directory Sources .....	33
Configuring the Sun Java System Directory Server Source .....	33
Configuring the Active Directory Source .....	35
Configuring the Windows NT Source .....	40
Configuring the Synchronization Settings .....	41
Configuring the Attribute Modification Settings .....	41
Configuring the Attributes Settings .....	42
Configuring the Object Creation Settings .....	43
Adding the ShadowAccount Objectclass .....	45
Configuring the Creation Attributes .....	45
Configuring the Synchronization User Lists .....	46
Installing the Connectors and Directory Server Plugin .....	57
Running idsync resync .....	57
Running Resynchronization Procedure When Directory Server is Authoritative .....	67
Configuration and Installation Summary .....	68
Multiple Domains .....	68
PAM LDAP .....	68
WAN Deployment .....	69
Migrating Users from Windows NT to Active Directory .....	70
Unlinking Migrated Windows NT Entries .....	71
Linking Migrated Active Directory Entries .....	75
Moving Users between Active Directory Organizational Units .....	76
When Contractors Change to Full-Time Employees .....	76
 <b>Chapter 3 Case Study: Deploying in a High-Availability Environment Over a Wide Area Network Using SSL .....</b>	<b>79</b>
Global Telco Deployment Information .....	80
Directory Server Setup .....	80
Active Directory Information .....	82
Requirements .....	83
Installation and Configuration Overview .....	84
Primary and Secondary Installations .....	84
Periodically Linking New Users .....	89
Large Deployment Considerations .....	89
Configuration Walkthrough .....	89
Primary Installation .....	90
Failover Installation .....	97
Setting Up SSL .....	101
Increasing Connector Worker Threads .....	106
Aligning Primary and Failover Configurations .....	108

Setting Multiple Passwords for uid=PSWConnector .....	109
Initial idsync resync Operation .....	111
Initial idsync resync Operation for Primary Installation .....	112
Initial idsync resync Operation for Failover Installation .....	115
Periodic idsync resync Operations .....	115
Periodic idsync resync Operation for Primary Installation .....	115
Periodic idsync resync Operation for Failover Installation .....	121
Configuring Identity Manager .....	122
Understanding the Failover Process .....	122
Directory Server Connector .....	123
Active Directory Connector .....	124
Initializing the Connector State .....	124
Failover Installation Maintenance .....	126
When to Failover .....	126
Failing Over .....	128
Stopping Synchronization at the Primary Installation .....	128
Starting Synchronization at the Failover Installation .....	129
Re-installing the Directory Server Plugins .....	129
Changing the PDC FSMO Role Owner .....	130
Monitoring the Logs .....	131
Failing Back to the Primary installation .....	131
<b>Appendix A Pluggable Authentication Modules .....</b>	<b>133</b>
Overview .....	134
Configuring PAM and Identity Synchronization for Windows .....	136
Step 1: Configure an LDAP Repository for PAM .....	136
Step 2: Configuring Identity Synchronization for Windows .....	139
Step 3: Populating the LDAP Repository .....	140
Step 4: Configuring a Solaris Host to Use PAM .....	144
Installing and Configuring a Solaris Test System .....	144
Configuring the Client Machine .....	144
Specifying Rules for Authentication and Password Management .....	146
Step 5: Verifying that PAM is Interoperating with the LDAP Store .....	150
Step 6: Demonstrating that User Changes are Flowing to the Reciprocal Environment .....	152
Case 1 .....	152
Case 2 .....	152
Case 3 .....	153
Case 4 .....	153
Configuring Systems to Prevent Eavesdropping .....	158
Introducing Windows NT into the System .....	158
Example /etc/pam.conf File .....	160

<b>Appendix B Identity Manager and Identity Synchronization for Windows Cohabitation</b>	<b>163</b>
Overview	164
Identity Manager and Identity Synchronization for Windows Functionality	165
Password Changes on Active Directory	166
Password Changes on Directory Server	166
Password Changes and Provisions Originating from Identity Manager Console	167
Configuring Identity Manager and Identity Synchronization for Windows	167
Setting Up Identity Manager 5.0 SP2 and Later	167
Configuring the Form Property	168
Configuring pwsync to Not Propagate Passwords to Directory Server	169
Setting Up Identity Manager 5.0 SP1 and Earlier	170
Configuring Identity Synchronization for Windows	171
Handling Identity Manager-Provisioned Users	172
<b>Appendix C Logging and Debugging</b>	<b>173</b>
Audit Logging and Action IDs	173
Actions	174
Connector Layers - Accessor, Controller, and Agent	175
Directory Server Plugin	180
Debug Logging	180
In Java Components	181
In the Installer	184
In the Console	185
Windows NT Change Detection	185
Changing Central Logs File Location	189
Changing Component Logs File Location	190
Isolating Problems in Directory Server	191
Isolating Problems in Message Queue	192
Isolating Problems in Active Directory	192
<b>Glossary</b>	<b>193</b>
<b>Index</b>	<b>205</b>

# List of Figures

Figure 2-1	Example Bank Architecture .....	24
Figure 2-2	Delegating User Control List Options .....	29
Figure 2-3	Selecting the Special User from the User and Group List .....	30
Figure 2-4	Tasks to Delegate Window Options .....	30
Figure 2-5	Objects to Delegate Control Wizard Options .....	31
Figure 2-6	Setting Permissions for the Special User Options .....	32
Figure 2-7	Preferred Directory Server Instance Dialog .....	34
Figure 2-8	Specifying the Secondary Master Directory Server .....	35
Figure 2-9	Windows Global Catalog Dialog Options .....	36
Figure 2-10	Credentials for the Active Directory Domain .....	37
Figure 2-11	PDC FSMO Role Owner Domain Controller Dialog .....	38
Figure 2-12	Failover Domain Controller Dialog Options .....	39
Figure 2-13	Windows NT Directory Source Selection Dialog Options .....	40
Figure 2-14	Attribute Mappings Window Options .....	41
Figure 2-15	Attribute Settings Window Options .....	42
Figure 2-16	Object Creation Settings Window Options .....	44
Figure 2-17	shadowAccount Object Class .....	45
Figure 2-18	Creation Attributes Dialog Options .....	46
Figure 2-19	Synchronization User List Dialog Options .....	47
Figure 2-20	Directory Server Criteria Options for Synchronization User List .....	48
Figure 2-21	Synchronization User List Dialog Box Options .....	49
Figure 2-22	Directory Server Criteria Options for Synchronization User List .....	50
Figure 2-23	Synchronization User List Dialog Options .....	51
Figure 2-24	Directory Server Criteria Options for Synchronization User List .....	51
Figure 2-25	Configuration Validity Status Dialog .....	52
Figure 2-26	A List of SUL Parameters for Directory Server .....	53
Figure 2-27	Validating SUL_AD_EAST .....	55

Figure 2-28	To Do List Dialog	56
Figure 2-29	Installation Steps Pending	57
Figure 3-1	Data Center Information for Directory Server	81
Figure 3-2	Data Center Information for Active Directory	82
Figure 3-3	Primary Installation of Identity Synchronization for Windows	85
Figure 3-4	Failover Installation while the Primary Installation is Active	87
Figure 3-5	Primary Installation after Reinstalling the Identity Synchronization for Windows Plugins	88
Figure 3-6	Configuring the Directory Server Source	90
Figure 3-7	Configuring the Directory Server Source Over SSL	90
Figure 3-8	Configuring Advanced Security Options for the Directory Server Source	91
Figure 3-9	Configuring the Active Directory Domain	92
Figure 3-10	Configuring Failover Active Directory Domain Controllers to Work over SSL	92
Figure 3-11	Security Option to Enable for the Active Directory Connector	93
Figure 3-12	Attribute Modification Flow Setting	93
Figure 3-13	Attribute Setting for Synchronization	94
Figure 3-14	Synchronization User List Creation	95
Figure 3-15	Excluding User from the Synchronization Process	95
Figure 3-16	Configuring the Preferred Directory Server	97
Figure 3-17	Configuring the Secondary Directory Server Master	97
Figure 3-18	Configuring the Active Directory Domain Controller	98
Figure 3-19	Warning Message	98
Figure 3-20	Configuring Domain Controllers for Failover during On-Demand Synchronization	99
Figure A-1	Summary of Configuration Screen	137
Figure A-2	Resulting Topology	139
Figure A-3	Creating a New User	141
Figure A-4	Enabling Posix User Attributes	141
Figure A-5	Sun ONE Directory Server Console	142
Figure A-6	Adding the shadowaccount Objectclass	143
Figure A-7	Example ldapclient Command	145
Figure A-8	Edited /etc/pam.conf File	148
Figure A-9	Creating a Directory	150
Figure A-10	Configured PAM LDAP System	150
Figure A-11	Auditing LDAP Operations	151
Figure A-12	Viewing the User on Windows	154
Figure A-13	George Washington's Properties	155
Figure A-14	Forcing a Password Change	156
Figure A-15	Change Password Dialog Box	156



Figure A-16	Providing a New Password .....	157
Figure A-17	Accepting the New Password .....	157
Figure B-1	Password Synchronization and User Creation in an Identity Manager-Identity Synchronization for Windows Environment .....	164
Figure C-1	Action Paths .....	176
Figure C-2	Log Configuration Tab .....	189



# List of Tables

Table 1	Typographic Conventions . . . . .	14
Table 2	Symbol Conventions . . . . .	15
Table 3	Shell Prompts . . . . .	16
Table 4	Books in This Documentation Set . . . . .	16
Table 2-1	List of Requirements . . . . .	26
Table 2-2	Example User Entry-SUL Matches . . . . .	53
Table 2-3	Guidelines for Handling Contractor Accounts . . . . .	77
Table 3-1	SSL Communication between Components . . . . .	104
Table C-1	Component-Level Debug Log Levels . . . . .	183



# Preface

The Sun Java™ System Identity Synchronization for Windows Deployment Planning Guide describes how to plan and implement the Sun Java™ System Identity Synchronization for Windows system.

This preface contains the following sections:

- Who Should Use This Book
- Before You Read this Book
- Conventions Used in this Book
- Related Documentation
- Accessing Sun Resources Online
- Contacting Sun Technical Support
- Related Third-Party Web Site References
- Sun Welcomes Your Comments

## Who Should Use This Book

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

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

# Before You Read this Book

Be sure you read the following Identity Synchronization for Windows publications before trying to deploy the Identity Synchronization for Windows product:

- *Sun Java™ System Identity Synchronization for Windows Installation and Configuration Guide.*
- *Sun Java™ System Identity Synchronization for Windows 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 Deployment Guide.

In addition, it is important that you read this Deployment Guide in its entirety before trying to deploy the Identity Synchronization for Windows product.

# Conventions Used in this Book

The tables in this section describe the conventions used in this book.

## Typographic Conventions

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

**Table 1**    Typographic Conventions

Typeface	Meaning	Examples
AaBbCc123 (Monospace)	API and language elements, HTML tags, web site URLs, command names, file names, directory path names, onscreen computer output, sample code.	Edit your <code>.login</code> file. Use <code>ls -a</code> to list all files. % You have mail.
<b>AaBbCc123</b> (Monospace bold)	What you type, when contrasted with onscreen computer output.	% <b>su</b> Password:

**Table 1** Typographic Conventions (*Continued*)

Typeface	Meaning	Examples
<i>AaBbCc123</i> (Italic)	Book titles, new terms, words to be emphasized.  A placeholder in a command or path name to be replaced with a real name or value.	Read Chapter 6 in the <i>Installation and Configuration Guide</i> .  These are called <i>class</i> options.  Do <i>not</i> save the file.  The file is located in the <i>install-dir</i> /bin directory.

## Symbols

The following table describes the symbol conventions used in this book.

**Table 2** Symbol Conventions

Symbol	Description	Example	Meaning
[ ]	Contains optional command options.	ls [-l]	The -l option is not required.
{   }	Contains a set of choices for a required command option.	-d {y n}	The -d option requires that you use either the y argument or the n argument.
-	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.

# Shell Prompts

The following table describes the shell prompts used in this book.

**Table 3** Shell Prompts

Shell	Prompt
C shell on UNIX or Linux	<i>machine-name%</i>
C shell superuser on UNIX or Linux	<i>machine-name#</i>
Bourne shell and Korn shell on UNIX or Linux	\$
Bourne shell and Korn shell superuser on UNIX or Linux	#
Windows command line	C:\

# Related Documentation

The <http://docs.sun.com><sup>SM</sup> web site enables you to access Sun technical documentation online. You can browse the archive or search for a specific book title or subject.

# Books in This Documentation Set

The following table summarizes the books included in the Identity Synchronization for Windows documentation set.

**Table 4** Books in This Documentation Set

Book Title	Description
<i>Sun Java System Identity Synchronization for Windows 1 2004Q3 Installation and Configuration Guide</i> ( <a href="http://docs.sun.com/app/docs/doc/817-6199">http://docs.sun.com/app/docs/doc/817-6199</a> )	Describes how to install and configure Identity Synchronization for Windows for use in a production environment.
<i>Sun Java System Identity Synchronization for Windows 1 2004Q3 Deployment Planning Guide</i> ( <a href="http://docs.sun.com/app/docs/doc/817-6200">http://docs.sun.com/app/docs/doc/817-6200</a> )	Provides general guidelines and best practices for the planning and deploying Identity Synchronization for Windows.
<i>Sun Java System Identity Synchronization for Windows 1 2004Q3 Release Notes</i> ( <a href="http://docs.sun.com/app/docs/doc/817-6202">http://docs.sun.com/app/docs/doc/817-6202</a> )	Available after the product is released. Contains last-minute information, including a description of what is new in this current release, known problems and limitations, installation notes, and how to report issues with the software or the documentation.



## Other Documentation

Because you will be working with Sun Java™ System Directory Server and Sun Java™ System Message Queue, you may need to refer to their product documentation. You can access the documentation from the following locations:

- Sun Java System Directory Server documentation  
[http://docs.sun.com/coll/DirectoryServer\\_04q2](http://docs.sun.com/coll/DirectoryServer_04q2)
- Sun Java System Message Queue documentation  
[http://docs.sun.com/app/docs/coll/MessageQueue\\_35\\_SP1](http://docs.sun.com/app/docs/coll/MessageQueue_35_SP1)

For information about the basic concepts of public-key cryptography; Secure Sockets Layer (SSL) protocol; intranet, extranet, and the 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 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*

## Accessing Sun Resources Online

For product downloads, professional services, patches and support, and additional developer information, go to the following:

- Product documentation on line  
<http://docs.sun.com>
- Product support and status  
<http://www.sun.com/service/support/software/>
- Sun Enterprise Services for Solaris patches and support  
<http://www.sun.com/service/>
- Developer information  
<http://www.sun.com/developers/>

- **Support and Training**  
<http://www.sun.com/supporttraining/>
- **Product data sheet**  
<http://www.sun.com/software/>

## Contacting Sun Technical Support

If you have technical questions about this product that are not answered in the product documentation, go to <http://www.sun.com/service/contacting>.

## Related Third-Party Web Site References

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

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.

## 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 document title and part number. The part number is a seven-digit or nine-digit number that can be found on the title page of the book or at the top of the document.

# Introduction

Sun Java™ System Identity Synchronization for Windows synchronizes user account information, including passwords, between Sun Java™ System 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.

The Identity Synchronization for Windows solution provides:

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

## Deployment Considerations

You must be aware of the following important deployment considerations before you begin deploying Identity Synchronization for Windows:

- **Synchronization direction(s) of passwords.**

If passwords are synchronized from Directory Server to Active Directory or in both directions, then you must install the High Encryption Pack on Windows 2000 to enable 128-bit SSL, which is required when setting passwords in Active Directory over LDAP.

- **Synchronizing the creation of new users.**

If Identity Synchronization for Windows does not synchronize the creation of new users, then you must run the `idsync resync` command periodically to establish links between newly created users. Changes to newly created users are not synchronized until the users are explicitly linked by running `idsync resync`. For details, see “Periodically Linking New Users” on page 89.

- **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 that must be run before synchronization is started. If more than 100,000 users are synchronized, then the `idsync resync` command should be run in batches for optimal performance, and to limit the load on Sun Java System Message Queue (Message Queue). For details, see “Periodically Linking New Users” on page 89.

- **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, then higher performance is achieved by distributing Identity Synchronization for Windows across multiple machines, for example, by installing 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, then the `activedirectorydomainname` (and/or `USER_NT_DOMAIN_NAME`) attributes should be synchronized to a Directory Server attribute. This is required to resolve ambiguity between Synchronization User List definitions. For details, see “Configuring the Synchronization User Lists” on page 46.

- **Number of Directory Server masters, hubs, and read-only replicas in the deployment.**

In a deployment with multiple Directory Servers, the Identity Synchronization for Windows Directory Server Plugin must be installed on each master, each hub, and each read-only replica. When configuring Identity Synchronization for Windows, one Directory Server master is designated as the preferred master. The Directory Server Connector detects and applies changes at the preferred master while it is running. If this server is down, then it can optionally apply changes at a second master. The Retro Changelog Plugin must be enabled on the preferred master, and this master should be on the same LAN as the Identity Synchronization for Windows Core.

- **Security.**

If the Directory Server or the Active Directory Connectors will connect to Directory Server or Active Directory over SSL, then SSL must be enabled in these directories. If the connectors are configured to only accept trusted certificates, then 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 Plugin and Active Directory, then SSL must be enabled in Directory Server, and the Certificate Authority certificate used to sign the Active Directory SSL certificate must be imported in the Directory Server's certificate database.



# Case Study: Deploying in an Multi-Master Replication Environment

The case study provided in this chapter explains how Example Bank implemented Identity Synchronization for Windows in an Multi-Master Replication (MMR) environment. The case study includes information about the business imperatives, the technical requirements, and the Identity Synchronization for Windows implementation.

This chapter contains these sections:

- Overview
- Deploying the Solution

## Overview

This section provides an overview of the architecture and technical requirements for “Example Bank” and the Identity Synchronization for Windows features used in this case study. The information is organized as follows:

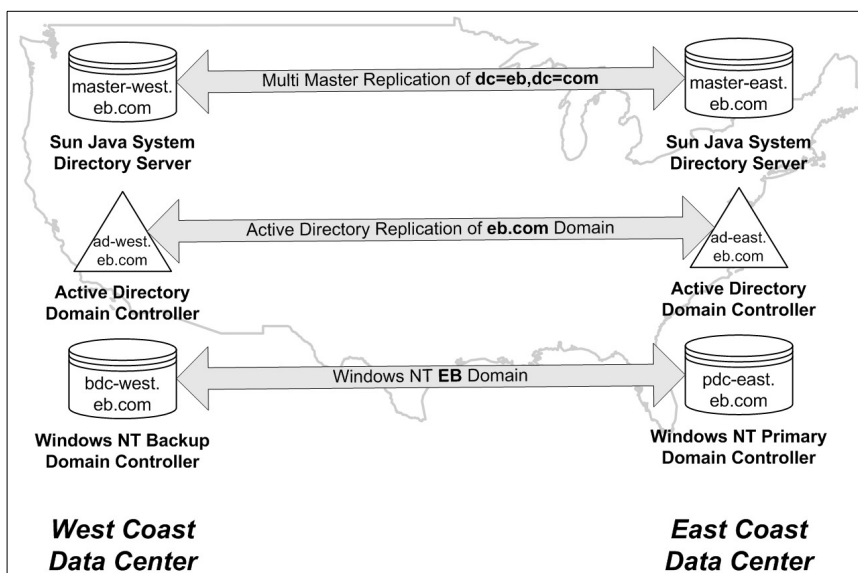
- “Existing Example Bank’s Architecture” on page 24
- “Understanding Example Bank’s Technical Requirements” on page 26
- “Identity Synchronization for Windows Features Discussed in this Case Study” on page 28

## Existing Example Bank's 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 Systems Directory Server (dc=eb,dc=com) deployment. They have two main sites; one located in New York city and other in Los Angeles.

The following figure describes Example Bank's deployment of their 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. Authentication and password management on Solaris is done against Directory Server using the Pluggable Authentication Module (PAM) for LDAP. The two directory server masters 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 9 machines, 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 PDC runs on pdc-east.eb.com machine which is located in New York (NY). A back-up domain controller (bdc-west.eb.com) runs on a machine located in Los Angeles (LA). All Windows NT users 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 Directory Server's `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)
- ad-west.eb.com (in Los Angeles)

In this deployment, ad-west.eb.com is the PDC-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, consequently, each employee has either a Windows NT or an Active Directory account. The migration of the users is based (in phases) on the employees' last names; each week they move users whose last name begins with the next letter of the alphabet. Currently, they have migrated employees whose last names begin with letters A through F.

For users that have Directory Server accounts, the Active Directory `samaccountname` attribute stores the `uid`. When a user 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.

# Understanding Example Bank’s Technical Requirements

The following table describes Example Bank’s technical requirements:

**Table 2-1** List of Requirements

Requirement	Comments
Users' Windows passwords should be synchronized with their Directory Server passwords.	Identity Synchronization for Windows can synchronize user's existing Directory Server password with their Active Directory password.
Users must be able to continue to change passwords using native mechanisms made in either environment.  That is, using the CTRL+ALT+DEL options on Windows and the passwd command on Solaris.	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).  For native Solaris password changes to be propagated to Active Directory, Solaris must be configured to use the Pluggable Authentication Module for LDAP. This is discussed in detail in Appendix A, "Pluggable Authentication Modules".  <b>Note:</b> Passwords can be set in the Sun Directory Server by passing in a pre-hashed password value. Identity Synchronization for Windows cannot synchronize passwords from Directory Server to Windows if the password is pre-hashed. This is normally prevented because it circumvents password policy and password history.
Contractors should not be synchronized. They might have accounts on Active Directory, Windows NT, or Directory Server but synchronization of their accounts should not be performed.  Contractors can be identified using their login name which starts with c-. The uid, samaccountname, and USER_NAME attributes will start with c- for the contractors only.	Synchronization User Lists (SUL) feature of Identity Synchronization for Windows can be used to eliminate user accounts that should not be synchronized. Part of an SUL's configuration includes an LDAP-like filter.  A contractor's login name begins with a c-.  To exclude contractors: <ul style="list-style-type: none"><li>Active Directory SUL filter will be <code>(!(samaccountname=c-*) )</code></li><li>Windows NT SUL filter will be <code>(!(USER_NAME=c-*) )</code></li><li>Directory Server SULs will include <code>(!(uid=c-*) )</code> as part of their filter. Directory Server SUL filters also include information about whether the user is on Windows NT or Active Directory</li></ul>
Changes to login IDs (USER_NAME, samaccountname, and uid) and full names (USER_FULL_NAME, cn, and cn) should be synchronized when they change.	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 even synchronize attributes that are otherwise used as keys such as uid and cn.  <b>Note:</b> The SUL filter applies to all operations, so all other changes for contractors will not be synchronized either.

**Table 2-1** List of Requirements

Requirement	Comments
When a new user is created in Active Directory (except for contractors), a corresponding user should be created in Directory Server.	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.
New users created in Directory Server should not have an Active Directory account created automatically.	Identity Synchronization for Windows can synchronize new user accounts in either direction from Windows to Directory Server or Directory Server to Windows (or not all).
Entries deleted in either Directory Server or Active Directory should not be deleted in the corresponding data source.	Identity Synchronization for Windows can synchronize account deletions from Active Directory to Directory Server, Directory Server to Active Directory, both ways, or not all. For this requirement, Identity Synchronization for Windows will not synchronize account deletions bi-directionally, that is, from Active Directory to Directory Server and from Directory Server to Active Directory.  <b>Note:</b> Identity Synchronization for Windows does not synchronize account deletions with Windows NT.
Accounts that are inactivated (disabled) in either Directory Server or Active Directory should be inactivated in the corresponding data source.	Identity Synchronization for Windows can synchronize inactivated accounts from Active Directory to Directory Server, Directory Server to Active Directory, both ways, or not all. For this requirement, Identity Synchronization for Windows synchronizes inactivated accounts bi-directionally, that is, from Active Directory to Directory Server and from Directory Server to Active Directory.  How Identity Synchronization for Windows detects account inactivations in Directory Server can be configured, Example Bank is not using a third-party application such as Sun Java System Access Manager to control access to their directory server, so the default account inactivation setting of inter-operating with the directory server tools suffices.  <b>Note:</b> Identity Synchronization for Windows does not synchronize account inactivations with Windows NT.
Once a user has been migrated from Windows NT to Active Directory, changes should be synchronized with the Active Directory account and not the Windows NT account.	Identity Synchronization for Windows stores a Globally Unique Identifier (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. By manually removing the attribute and then re-running the <code>idsync resync -f &lt;linking-file&gt;</code> command, the entry is updated in Directory Server.
Users can move between <code>ou=east</code> and <code>ou=west</code> .	Identity Synchronization for Windows can move users between the Synchronization User Lists (SUL), and these changes are performed automatically without any administrator intervention.

## Identity Synchronization for Windows Features Discussed in this Case Study

The following Identity Synchronization for Windows features are used for 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
- Integration with PAM LDAP
- Moving users between Windows domains, specifically from Windows NT to Active Directory
- Excluding users from synchronization using an SUL filter
- Synchronizing hierarchical DIT with a flat DIT
- Creating attribute default values

## 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 33
- “Migrating Users from Windows NT to Active Directory” on page 70

## Creating a Special Active Directory User for Identity Synchronization for Windows

Example Bank first creates a special user that Identity Synchronization for Windows uses when connecting to Active Directory. This user is first created in the `cn=Users` container in the `eb.com` domain. Once 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.

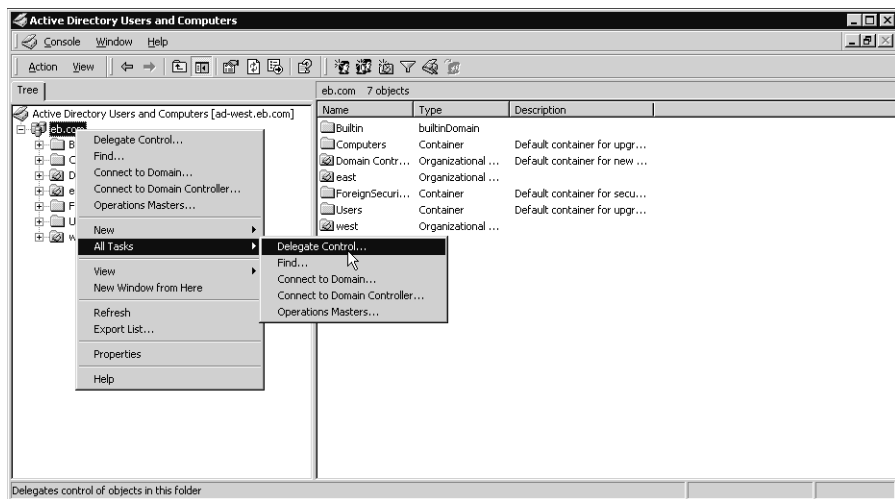
---

➤ **To assign administration rights to this special user**

1. In the Active Directory Users and Computers domain, from the Tree pane, right-click on the `eb.com` container icon.
2. Select **All Tasks > Delegate Control** from the list.

The Delegation of Control Wizard displays.

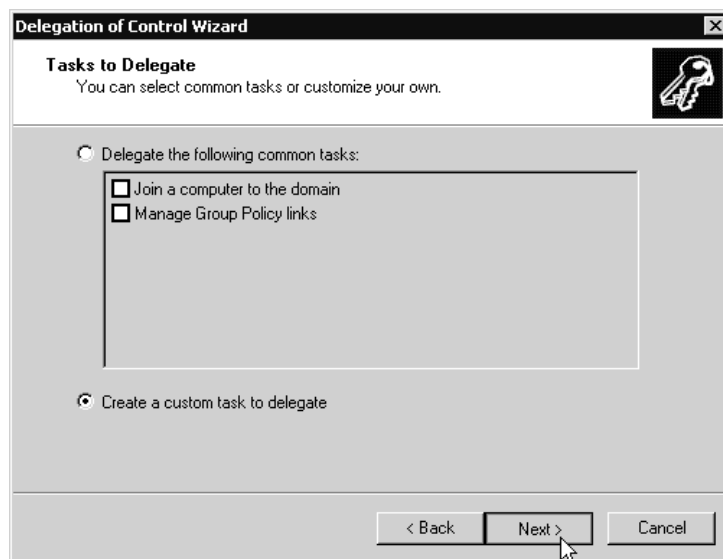
**Figure 2-2** Delegating User Control List Options



3. Select the special user from the “Selected user and groups” list, and click Next.

**Figure 2-3** Selecting the Special User from the User and Group List

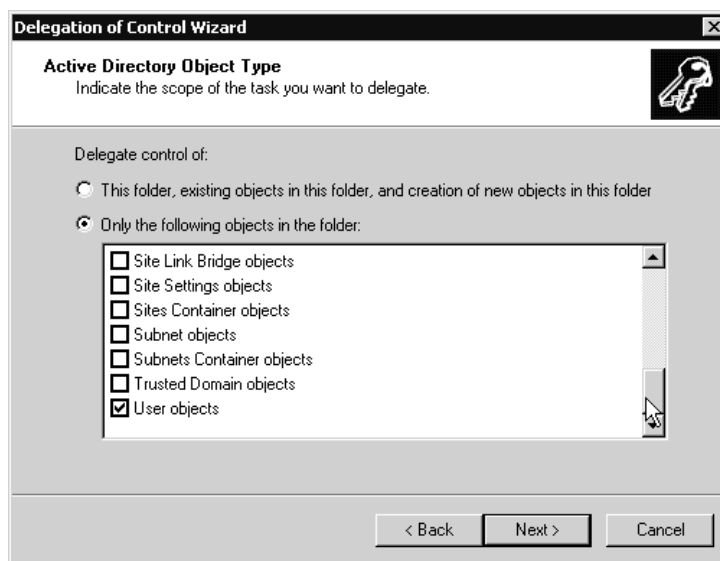
4. Choose the Create a custom tasks to delegate option from the “Tasks to Delegate” window, and click Next.

**Figure 2-4** Tasks to Delegate Window Options

5. Select the User Objects option from the list in the Only the following objects in the folder section.

Because Identity Synchronization for Windows only manages User objects, it is sufficient to delegate control of these objects only.

**Figure 2-5** Objects to Delegate Control Wizard Options



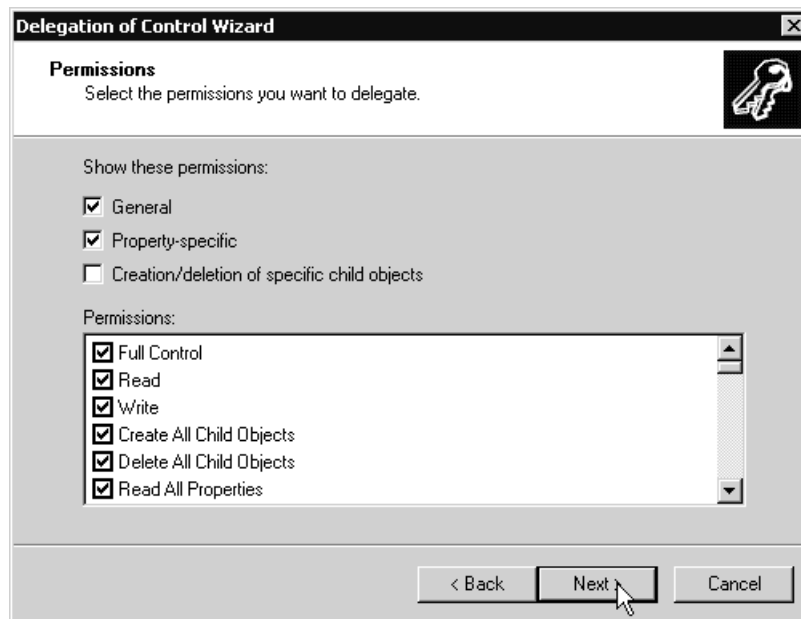
6. In the Permissions window, from the Show these permissions list, ensure that you select these options:
  - General
  - Property-Specific
  - Full Control

Because Example Bank wants to synchronize 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 (such as an `idsync resync` operation from Active Directory to Directory Server) will succeed, but other operations (such as detecting and applying changes in Active Directory) can fail abruptly.

If Example Bank is synchronizing deletes from Active Directory to Directory Server, then even Full Control is insufficient. You must Identity Synchronization for Windows to use a Domain Administrator account to detect account deletions in Active Directory.

**Figure 2-6** Setting Permissions for the Special User Options





## 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 Directory Server master is running). Once the Core is installed, complete the following configuration procedure.

- “Configuring Directory Sources”
- “Configuring the Synchronization Settings”
- “Configuring the Synchronization User Lists”
- “Installing the Connectors and Directory Server Plugin”
- “Running idsync resync”
- “Configuration and Installation Summary”

## Configuring Directory Sources

This section explains how to configure the following directory sources:

- “Configuring the Sun Java System Directory Server Source” on page 33
- “Configuring the Active Directory Source” on page 35
- “Configuring the Windows NT Source” on page 40

### Configuring the Sun Java System Directory Server Source

When configuring the Directory Server source, the preferred directory server instance is set to master-east.eb.com. The Directory Server Connector uses this instance for detecting changes that require synchronization with the Active Directory and Windows NT, and also for updating the changes that are detected in the Active Directory and Windows NT. Alternatively, the master-west.eb.com domain could have been selected; however, the Directory Server Connector performance improves when connecting to a local directory server instead of a Directory Server located over a wide area network (WAN).

---

<b>NOTE</b>	Although SSL is not selected originally, it is required because Example Bank is synchronizing from Directory Server to Active Directory. The console will automatically enable this option when the password modification settings are changed.
-------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

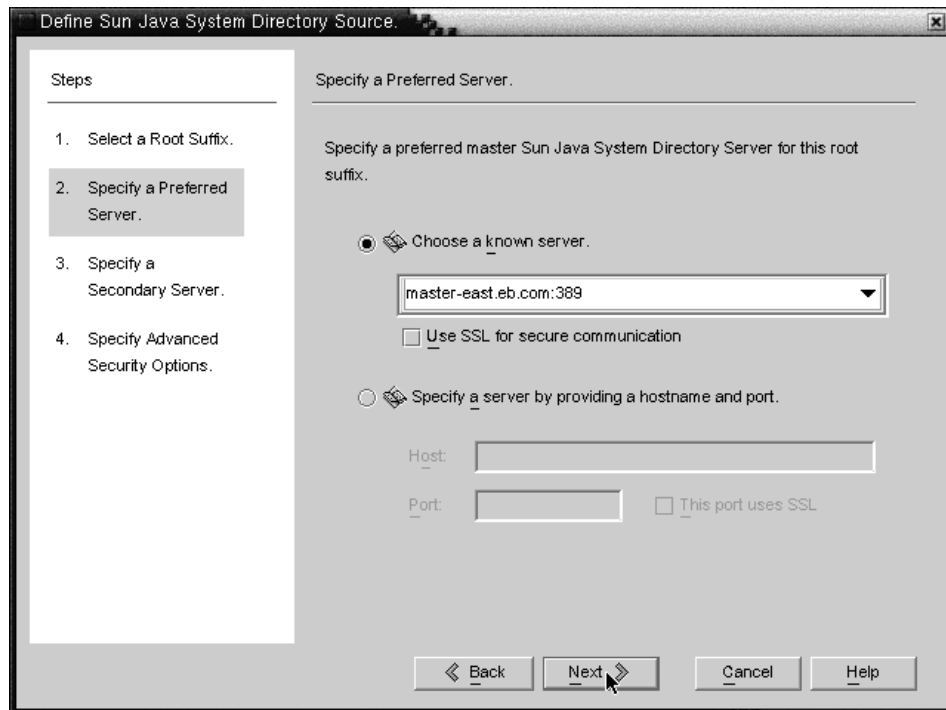
---

➤ **To specify a preferred directory source**

1. In the Console, from the Directory Sources window, click New Sun Directory.

In the Specify a Preferred server, from the Define Sun Java System Directory Source dialog, select the appropriate Directory Source. In this case, master-east.eb.com

**Figure 2-7** Preferred Directory Server Instance Dialog

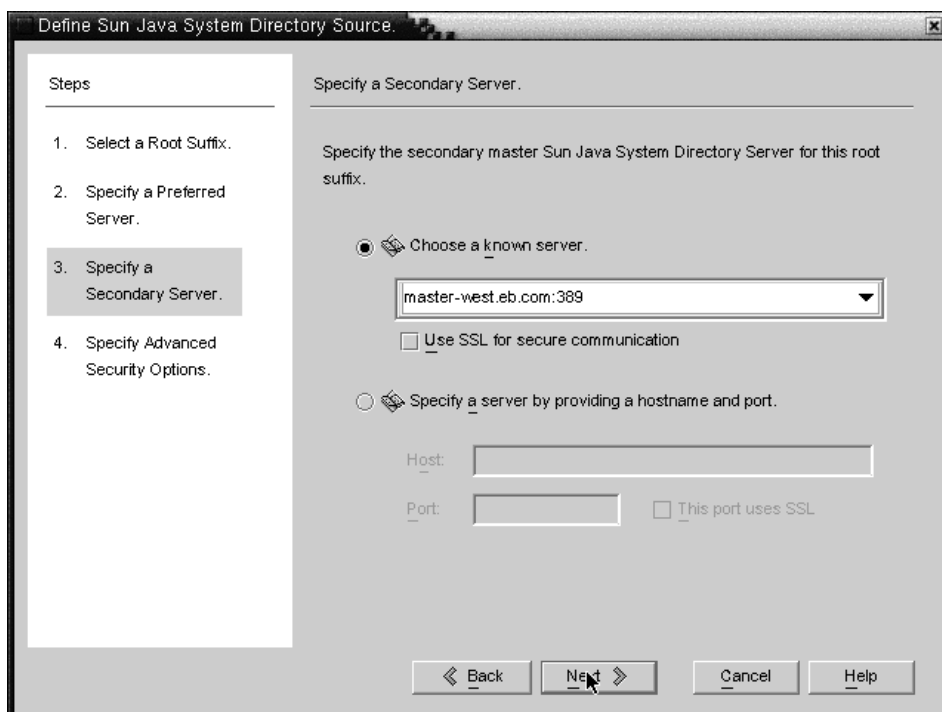


2. In the Specify a Secondary Server, select the secondary Directory Server instance. In this case, master-west.eb.com.

A secondary Directory Server instance can be specified to ensure availability when the preferred master, master-east.eb.com, is unavailable. The Secondary Directory Server instance, master-west.eb.com is selected as the secondary master directory server.

If master-east.eb.com is unavailable, then the Directory Server Connector synchronizes changes made at the Active Directory to master-west.eb.com. Configuring a secondary directory source is optional.

**Figure 2-8** Specifying the Secondary Master Directory Server



## Configuring the Active Directory Source

The Active Directory Global Catalog information is provided to allow the Identity Synchronization for Windows console to discover 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=users,dc=eb,dc=com`, but this is changed 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**

1. In the Console, from the Directory Sources window, click New Active Directory Source button.

The Windows Global Catalog dialog displays.

2. Enter the fully qualified name in the Host field. In this example, it's ad-west.ed.com.
3. Change the default User DN (cn=Administrator) to the DN shown below as described earlier.
4. Enter the password and click OK.

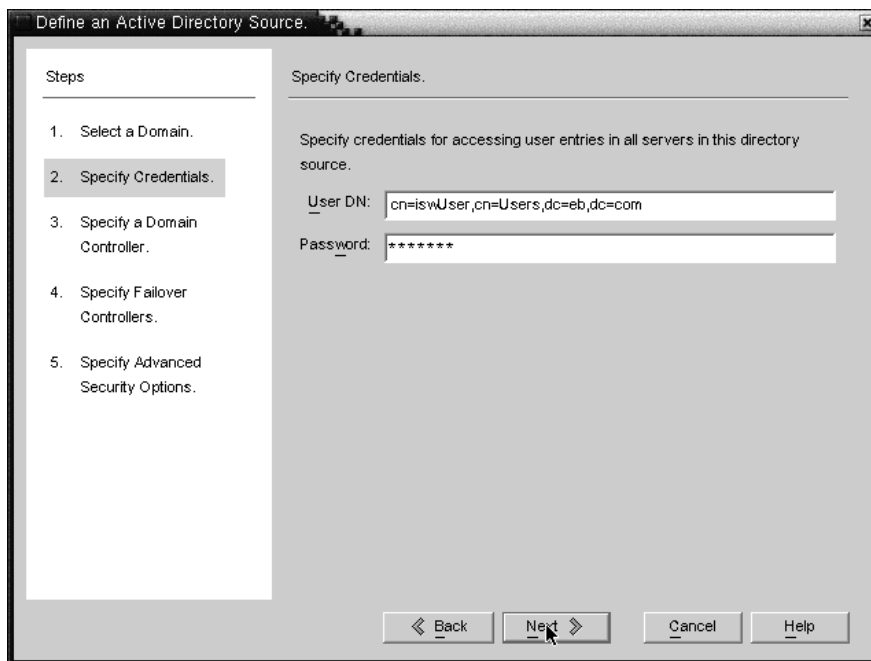
**Figure 2-9** Windows Global Catalog Dialog Options



5. Provide credentials for the Active Directory domain.

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.

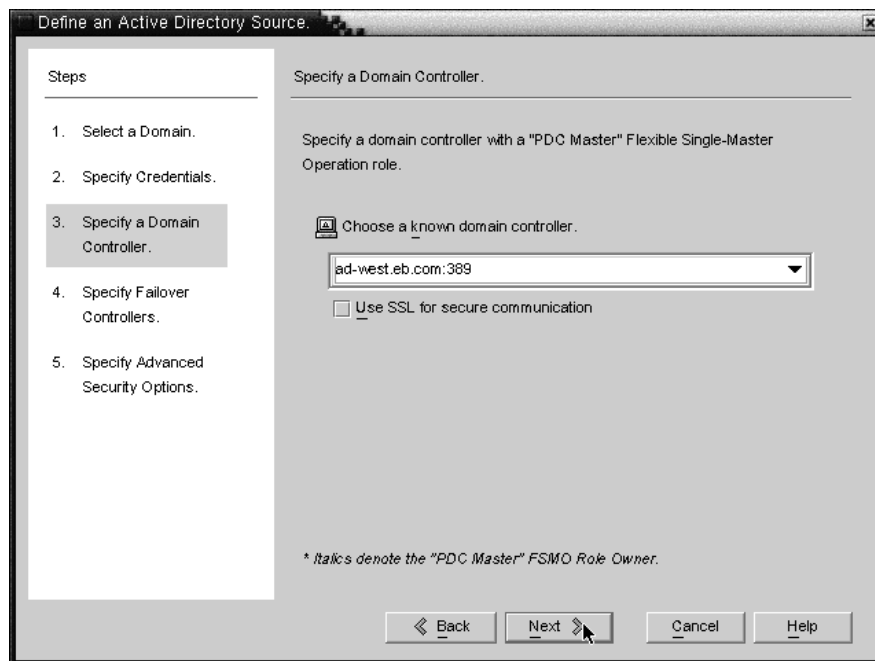
**Figure 2-10** Credentials for the Active Directory Domain



#### 6. Specify the PDC FSMO role owner domain controller.

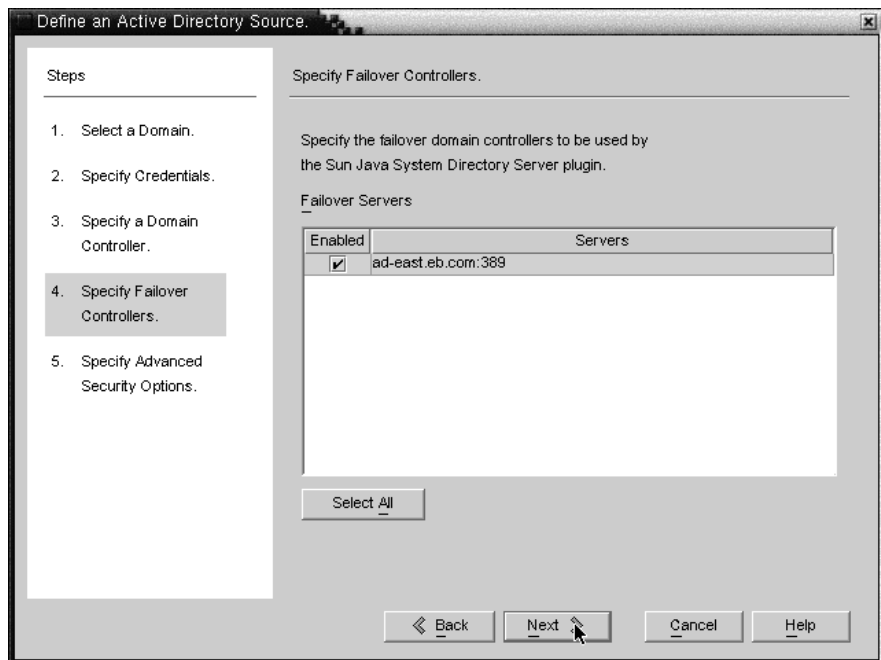
The ad-west.eb.com domain controller is the PDC FSMO role owner and certain changes (for example, password modifications) made at other domain controllers are replicated immediately to it. The connector communicates with this domain controller so that changes made at any Active Directory domain controller 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. Because the Active Directory Connector does fewer directory searches during change detection, the Active Directory Connector performs better across the WAN than the Directory Server Connector.

**Figure 2-11** PDC FSMO Role Owner Domain Controller Dialog

**7. Specify the failover domain controller for on-demand password synchronization.**

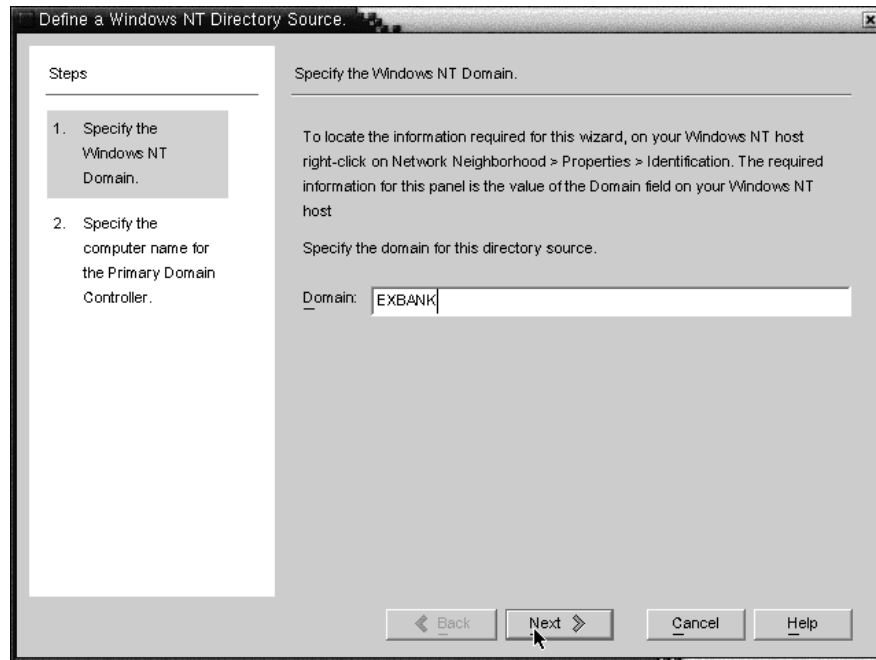
The ad-east.eb.com domain controller is selected as a failover domain controller for on-demand password synchronization. If ad-west.eb.com is unavailable, then the Directory Server Plugin performs on-demand password synchronization against ad-east.eb.com.

**Figure 2-12** Failover Domain Controller Dialog Options

## Configuring the Windows NT Source

After the Directory Server and the Active Directory sources are configured, information is provided for the Windows NT domain.

**Figure 2-13** Windows NT Directory Source Selection Dialog Options



These details are specified for the Windows NT directory source:

- The Windows NT domain name is specified. EXBANK is the name of the Windows NT domain that is synchronized.
- 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

Once each directory source is configured, the synchronization parameters are configured to match Example Bank's requirements.

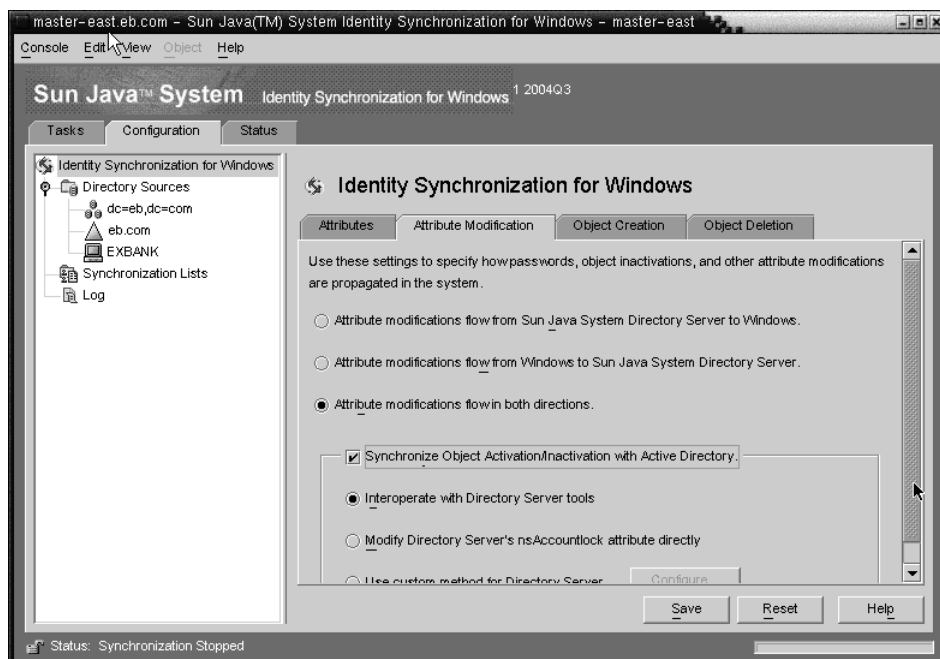
### Configuring the Attribute Modification Settings

These Attribute Modification settings reflect that Example Bank's requirements to synchronize attribute changes and account inactivations, bi-directionally, between the Active Directory and Directory Server sources.

#### ► To configure the attribute mappings

1. In the Console, select the Configuration tab.
2. Select the Attribute Modification tab.
3. Select the "Attribute modifications in both directions" option.
4. Select the "Synchronize Object Activation/Inactivation with Active Directory" option and, also ensure you select the "Interoperate with Directory Server tools" options.

**Figure 2-14** Attribute Mappings Window Options



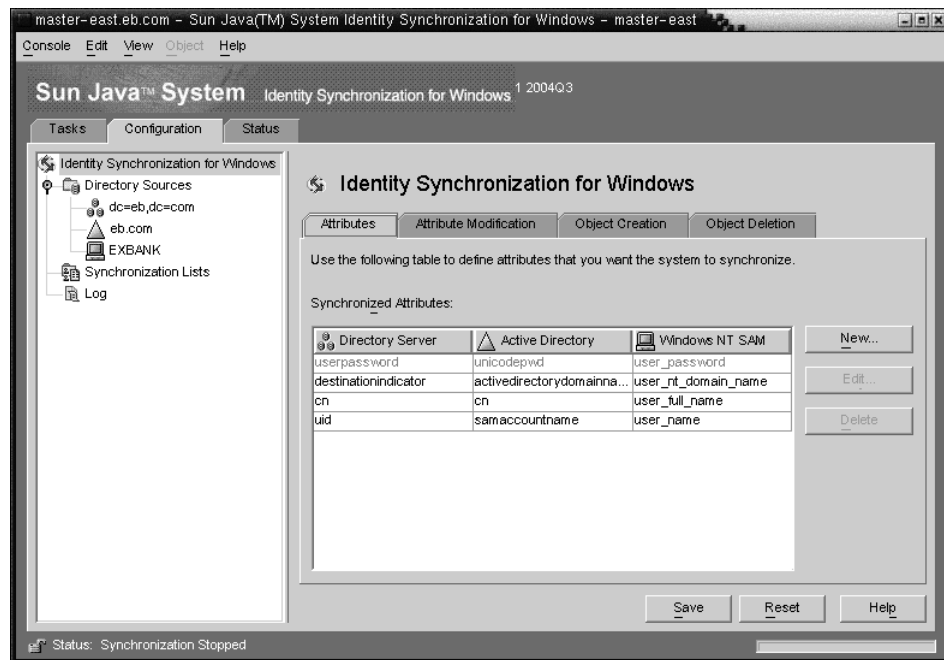
## Configuring the Attributes Settings

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, select the Configuration tab.
2. Select the Attributes tab.
3. In the Synchronized Attributes list, enter the attributes that Example Bank's requires synchronization with Directory Server.

**Figure 2-15** Attribute Settings Window Options



---

**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`.

---

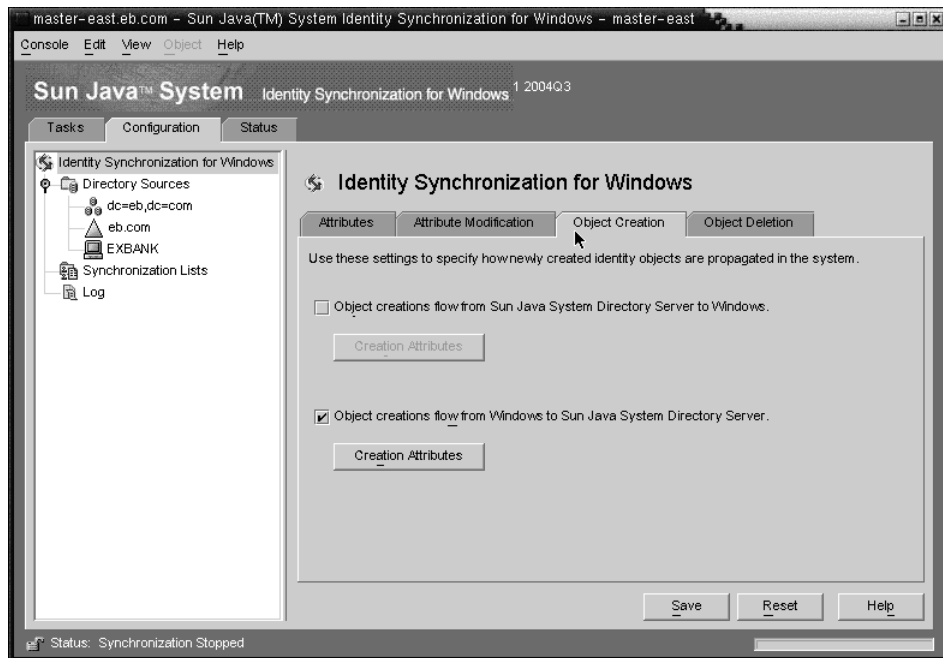
## Configuring the Object Creation Settings

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

Because Example Bank has an environment with both Active Directory and Windows NT, these settings apply to both systems. Not only will new users in Active Directory be synchronized to Directory Server, but also any new user created in Windows NT will also be synchronized. Because Example Bank is migrating all Windows NT users to Active Directory, no new users will be created in Windows NT, and this does not present a problem.

### ► To configure the attribute mappings

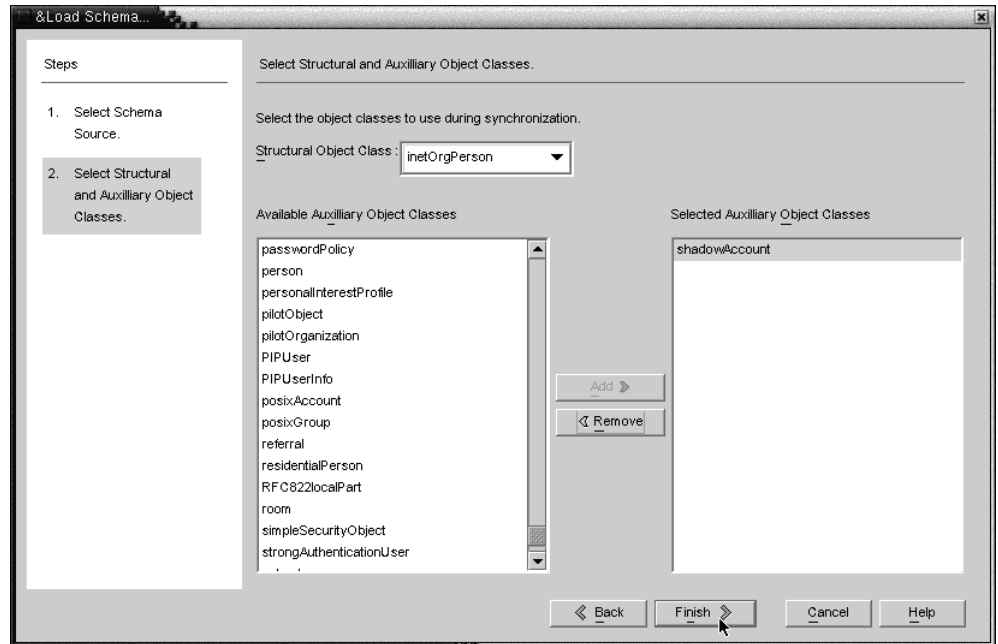
1. In the Console, select the Configuration tab.
2. Select the Object Creation tab.
3. Select the "Object Creations flow from Windows to Sun Java System Directory Server" option.

**Figure 2-16** Object Creation Settings Window Options

## Adding the ShadowAccount Objectclass

When configuring Identity Synchronization for Windows to interoperate with Solaris PAM LDAP, the `shadowAccount` object class is chosen 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 2-17** shadowAccount Object Class

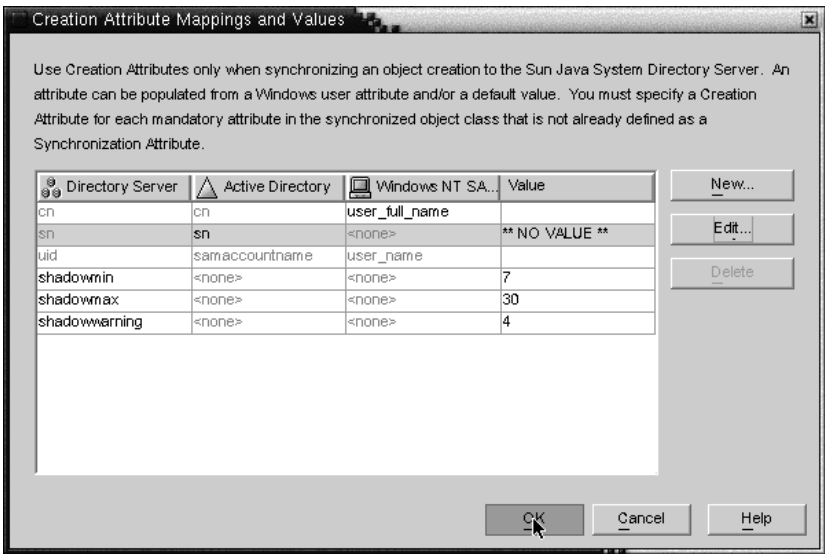


## Configuring the Creation Attributes

Use the Creation Attributes Mappings and values dialog box to configure additional attributes to be synchronized when an entry is created. Because `sn` is a mandatory attribute for the `inetOrgPerson` objectclass, you must provide a mapping or default value for `sn`. 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 and is only provided to conform to the Console's validations.

The shadowmin, shadowmax, and shadowwarning attributes are used for PAM LDAP. A shadowmin value of 7 implies that a user must wait seven days once 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 shown in grey are mandatory creation attributes. The inetorgperson objectclass has cn and sn as mandatory attributes, and the shadowaccount objectclass has uid as a mandatory attribute.

**Figure 2-18** Creation Attributes Dialog Options



## Configuring the Synchronization User Lists

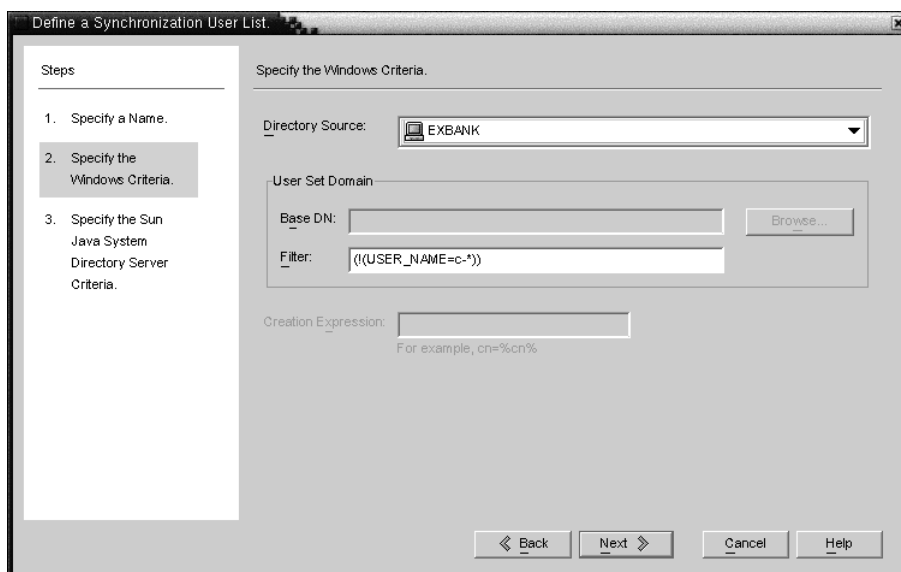
Because Example Bank has two Windows domains (one for Windows NT and one for Active Directory), at least two Synchronization User Lists (SUL) are required; one for Active Directory and one for Windows NT. However, Example Bank wants to synchronize two disjoint organizational units in Active Directory (ou=east and ou=west), and 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 contractors and only contractors' `USER_NAME` value starts with `c-`.

**Figure 2-19** Synchronization User List Dialog Options



The base DN of all synchronized users is “`ou=people,dc=eb,dc=com`” and 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.

**Figure 2-20** Directory Server Criteria Options for Synchronization User List

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=people,dc=eb,dc=com  
For example, cn=%cn%

### *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.

These 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).



**Figure 2-21** Synchronization User List Dialog Box Options

The screenshot shows a Windows-style dialog box titled "Define a Synchronization User List". On the left, there's a sidebar with three steps: "Specify a Name.", "Specify the Windows Criteria." (which is highlighted), and "Specify the Sun Java System Directory Server Criteria.". The main area is titled "Specify the Windows Criteria." and contains several input fields. There's a "Directory Source:" dropdown menu showing "eb.com". Below it is a section labeled "User Set Domain" containing a "Base DN:" field with "OU=east,DC=eb,DC=com", a "Filter:" field with "(!(&samaccountname=c\*))", and a "Browse..." button. At the bottom, there's a "Creation Expression:" field with "OU=east,DC=eb,DC=com" and a note "For example, cn=%cn%". Navigation buttons at the bottom include "< Back", "Next >", "Cancel", and "Help". A mouse cursor is pointing at the "Next >" button.

For SUL\_NT list, only users under the ou=people,dc=eb,dc=com organizational unit will be synchronized. The filter for this SUL includes two components:  
(!(uid=c-\*)) excludes contractors from synchronization, and  
(destinationindicator=eb.com) allows the connector to determine the Windows domain where the user resides.

**Figure 2-22** Directory Server Criteria Options for Synchronization User List

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:  ,dc=eb,dc=com  
For example, cn=%cn%

### *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.

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

**Figure 2-23** Synchronization User List Dialog Options

The dialog box is titled "Define a Synchronization User List". On the left, a "Steps" pane shows three steps: "1. Specify a Name.", "2. Specify the Windows Criteria." (which is highlighted), and "3. Specify the Sun Java System Directory Server Criteria.".

The main area is titled "Specify the Windows Criteria." and contains the following fields:

- Directory Source:** A dropdown menu showing "eb.com".
- User Set Domain:** A section containing:
  - Base DN:** A text field with "ou=west,dc=eb,dc=com" and a "Browse..." button.
  - Filter:** A text field with "(!(&samaccountname=c\*))".
- Creation Expression:** A text field with "ou=west,dc=eb,dc=com". Below it, a note says "For example, cn=%cn%".

At the bottom, there are four buttons: "Back", "Next", "Cancel", and "Help".

These settings are identical to the `SUL_AD_EAST` synchronization user list, which causes the following error.

**Figure 2-24** Directory Server Criteria Options for Synchronization User List

The dialog box is titled "Define a Synchronization User List". On the left, a "Steps" pane shows three steps: "1. Specify a Name.", "2. Specify the Windows Criteria.", and "3. Specify the Sun Java System Directory Server Criteria." (which is highlighted).

The main area is titled "Specify the Sun Java System Directory Server Criteria." and contains the following fields:

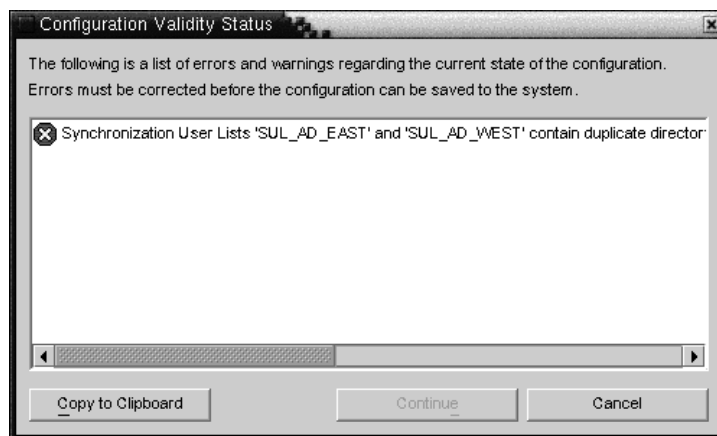
- Directory Source:** A dropdown menu showing "dc=eb,dc=com".
- User Set Domain:** A section containing:
  - Base DN:** A text field with "ou=people,dc=eb,dc=com" and a "Browse..." button.
  - Filter:** A text field with "(&!(uid=c-\*)(destinationIndicator=eb.com))".
- Creation Expression:** A text field with "uid=%uid%". Below it, a note says "For example, cn=%cn%".

At the bottom, there are four buttons: "Back", "Finish", "Cancel", and "Help".

When the configuration is saved, this 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)).”

**Figure 2-25** Configuration Validity Status Dialog

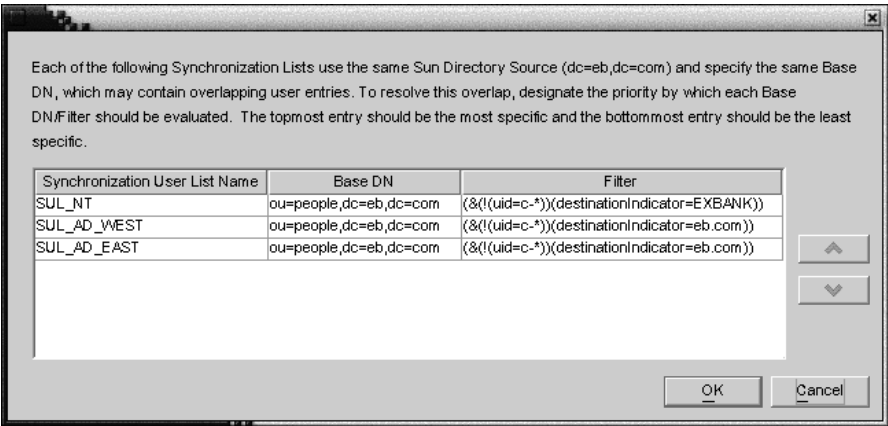


When there are multiple SULs, Identity Synchronization for Windows requires that the base DN or the filter is unique. When Identity Synchronization for Windows detects a change to a user, it iterates through the list of SULs until it matches a user.

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

When a change to a user is detected, that user entry is compared first with SUL\_NT, if it fails to match, it's compared with SUL\_AD\_WEST, and if it fails to match again, it's compared with SUL\_AD\_EAST.

Figure 2-26 A List of SUL Parameters for Directory Server



For example:

Table 2-2 Example User Entry-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-*
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 dns

Because SULs are evaluated from top to bottom, the program will not find a match for the SUL\_AD\_EAST entry. The program will encounter and evaluate the base dn and filter for the SUL\_AD\_WEST entry first, which is located immediately before SUL\_AD\_EAST entry. Consequently, the Console requires SULs to have different base DNs or filters.

An SUL is used primarily when a change to a user is detected. Evaluating which SUL the user resides in determines which 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 it will only link to a Directory Server user that is in the same SUL as the Active Directory user. This can be overridden as described in “Running `idsync resync`” on page 57.
- 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.

Because the default behavior of `idsync resync` can be overridden and because Example Bank does want to synchronize user creations from Directory Server to Active Directory, the `SUL_AD_WEST` and `SUL_AD_EAST` can have the same base `dn` and filter.

You can ignore this warning message and, as a workaround, you can rearrange the filters. Change the `(&(!(uid=c-*)(destinationindicator=eb.com))` filter to be equivalent to `(&(destinationindicator=eb.com)(!(uid=c-*)`). Because the Console only does string comparisons to determine if two filters are equivalent, this rearrangement passes the console’s validations.

---

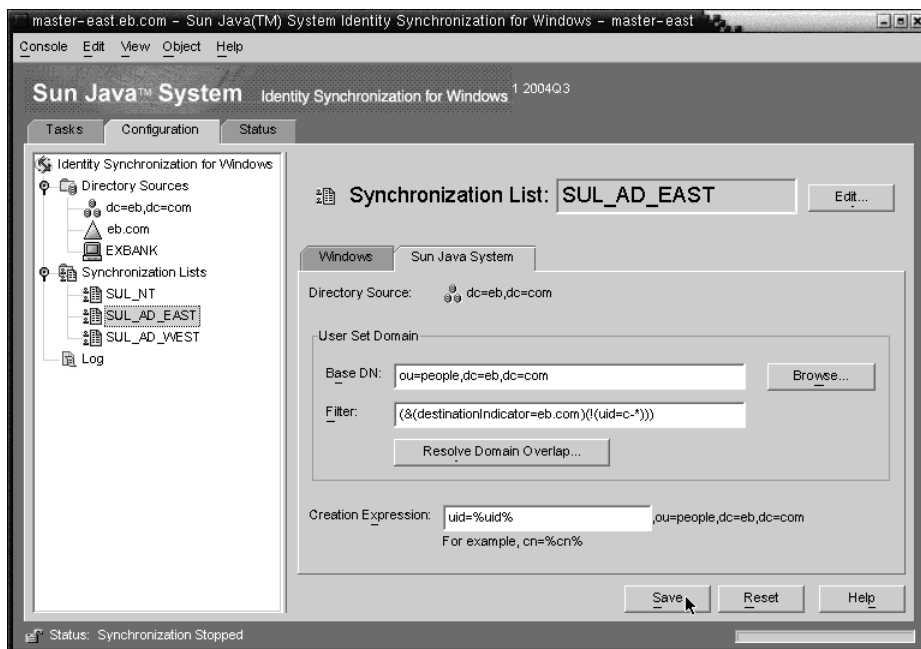
**NOTE** If there is no SUL filter or it only has a single component, then an SUL filter (or filter component) is added. This SUL filter component is different for each SUL but always evaluates to true.

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

```
(&(!(uid=c-*)(destinationIndicator=eb.com)
  (!(bogusAttr=SUL_AD_WEST)))

(&(!(uid=c-*)(destinationIndicator=eb.com)
  (!(bogusAttr=SUL_AD_EAST)))
```

---

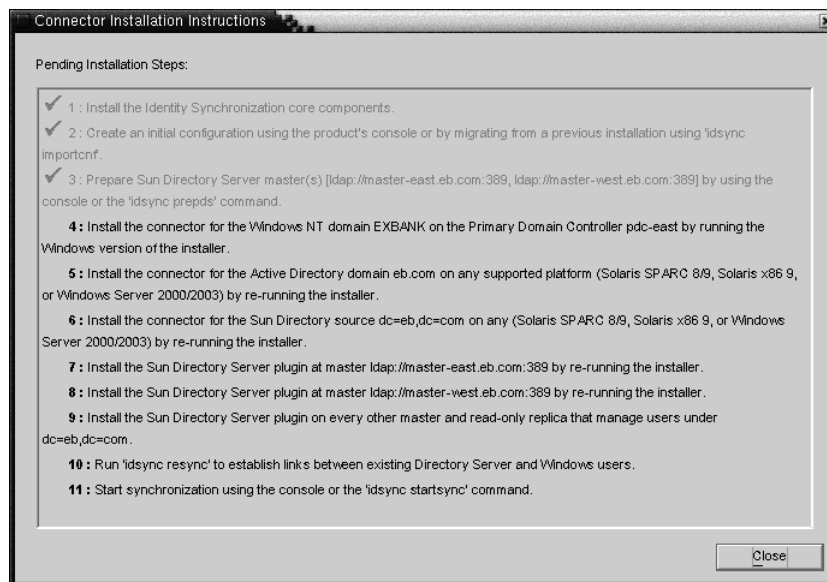
**Figure 2-27** Validating SUL\_AD\_EAST

The new definition of SUL\_AD\_EAST passes the Console's validation.

### *TODO List*

Once the configuration is saved successfully, the following list of pending installation steps should be completed.

**Figure 2-28** To Do List Dialog

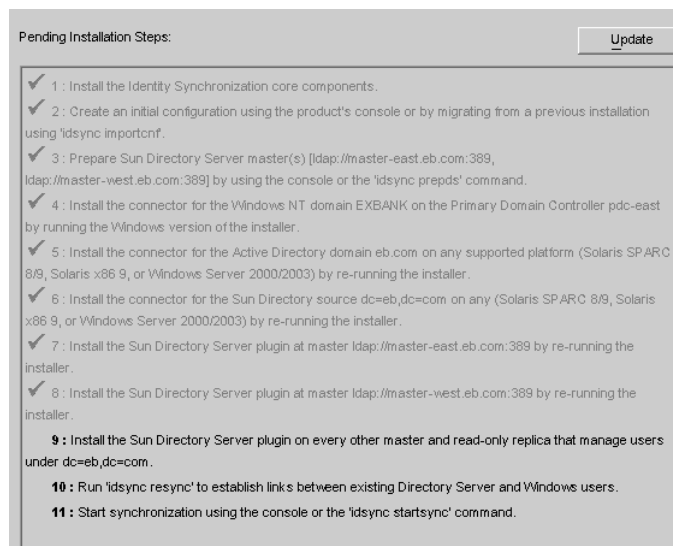




## Installing the Connectors and Directory Server Plugin

The Active Directory and Directory Server connectors are installed on master-east.eb.com. The Windows NT Connector is installed on the PDC pdc-east.eb.com. The Directory Server Plugins are installed on the two masters master-east.eb.com and master-west.eb.com.

**Figure 2-29** Installation Steps Pending



Because there are no other directory server masters or read-only replicas in this environment, the next step is to perform the `idsync resync` procedure to link all existing users.

## Running idsync resync

In Example Bank's deployment, users already have accounts in Directory Server and in Windows, so you must run `idsync resync` to establish links between equivalent users before starting synchronization. Linking the users is done using the `-f <linking-file>` option to `resync`. For detailed information on the `idsync resync` command, see the *Sun Java System Identity Synchronization for Windows Installation and Configuration Guide*.

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

For all users with Active Directory accounts, their Directory Server password 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:

```
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:: AQUAAAAAAAAUVAAAFSWvR+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:

```

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

```

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

For information on the syntax used in this example, see the *Sun Java System Identity Synchronization for Windows Installation and Configuration 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>
</UserLinkingOperationList>

```

```

    </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` 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 executed again with the new `linkusers-ad-only.cfg` linking file and the `-a` option to run the command for our test user John Test only. However, a message is displayed that indicates one entry matched a user in Directory Server but the Directory Server user was not 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"` option to the is added to the `linkusers-ad-only.cfg` file:

---

<b>NOTE</b>	Whenever a configuration has multiple SULs, the <code>allowLinkingOutOfScope=true</code> parameter should be used.
-------------	--------------------------------------------------------------------------------------------------------------------

---

```
<?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.
- Although the Directory Server password was reset to the Active Directory value, the `userPassword` attribute has not changed. (The value does not change until the user logs onto Directory Server using the Active Directory password and triggers on-demand password synchronization.)
- The user now has the `dspswuser` auxiliary objectclass. This objectclass 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}sTpxX8RQcz4GjqJOttSauXNjWcnaR/hCLX7gPA==
uid: jtest
givenName: John
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
objectClass: dspswuser
sn: Test
```

Verify that John Test can log onto 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}8wmyeFe2bLrOkwM/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 with `linkusers-nt-only.cfg`, 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 their users by running `idsync resync`, most of the users are linked successfully, but some users cannot be linked due to data inconsistencies. Once these inconsistencies are manually repaired, `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 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 above examples, the `destinationIndicator` attribute is automatically populated with the correct Windows domain name. Because they are synchronized attributes, the Directory Server attributes `cn` and `uid` are also updated.

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

The following procedure describes synchronization of attribute values in Active Directory with the values in Directory Server after linking the entries.

- Change the configuration such that only synchronized attributes are `userPassword` and `destinationIndicator`.
- Execute the `idsync resync -f <linking file>` command to link the entries and populate the `destinationIndicator` attribute
- Change the configuration to include all synchronized attributes (for example, add `cn` and `uid`).
- 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, then 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 procedure as determined by the main requirements of Example Bank.

### Multiple Domains

When configuring Identity Synchronization for Windows to support multiple domains:

- Set up `destinationIndicator` <-> `activedirectorydomainname` <-> `user_nt_domain_name` as a synchronized attribute.
- Use `destinationIndicator` in the SUL filters so that entries modified in Directory Server can be located in the proper Active Directory domain.
- When linking users using `idsync resync` command, specify `allowLinkingOutOfScope="true"` in the input file. Do not specify the `-k` option so that the `destinationIndicator` attribute can be primed.

### PAM LDAP

This section describes the procedure necessary to configure Identity Synchronization for Windows to support PAM LDAP.

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

For information on the prerequisites and details on the procedure to conform to PAM LDAP on the Solaris side, see Appendix A.

## WAN Deployment

Identity Synchronization for Windows has limited support for WAN deployments. It can synchronize with the Directory Server or the 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.

This was achieved in this scenario by installing the Identity Synchronization for Windows Core, Directory Server Connector, and the Active Directory Connector on the same machine, and the 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 will be significantly delayed if it was selected.

---

**NOTE** When the Directory Server and Active Directory domain controller are separated by a WAN, you have the option of installing Identity Synchronization for Windows:

- On the same LAN as Directory Server
- Or on the same LAN as Active Directory
- Or 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 requires at least 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 discusses the procedure to migrate users from Windows NT to Active Directory (moving a user between Windows domains).

- Moving User Accounts from Windows NT to Active Directory
- Unlinking the 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 their last name and that they have migrated users whose last names start with A-F.

Once 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, the entry will still be linked to a Windows NT entry. This entry is processed by the Windows NT Connector to process but the user no longer exists in Windows NT.
- Changes made in Active Directory also will be detected but the entry will match no 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 for the migrated Windows NT accounts, and then the Active Directory accounts are linked to their corresponding Directory Server entries.

---

<b>NOTE</b>	To avoid losing too many changes, the full migration process should be done when the load on Directory Server is light. The <code>idsync resync</code> command can be run to recover any change except changes to the passwords.
-------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

In this example, the users are moved from a Windows NT domain to a Windows 2000 domain, but the same procedure can be followed when moving users are between Windows NT domains or Windows 2000 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. It 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 $ldif = shift @_;
    my @dns = ();
    # Note that DNS can span multiple lines.
    while ($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: dspwuserlink\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. Once these users have been migrated, the following `unlink.pl` script invocation 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 subset of users.

## Linking Migrated Active Directory Entries

Once the links in the Directory Server entries are removed, new links are established with the Active Directory entries. The `idsync resync` command is executed to achieve this. 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.

If any of the users' passwords are modified in Directory Server during the migration process, these 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), nothing needs to be done for this user to continue to be synchronized by Identity Synchronization for Windows.

## When Contractors Change to Full-Time Employees

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

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

**Table 2-3** Guidelines for Handling Contractor Accounts

Active Directory	Directory Server	Creating Linked Entries in Active Directory and Directory Server
No	No	This may not occur because contractors have either an Active Directory or Directory Server account. If it does occur, then create a new entry in Active Directory, and Identity Synchronization for Windows automatically creates a new user in Directory Server
No	Yes	<ol style="list-style-type: none"> <li>1. Remove the <code>c-</code> prefix from the Directory Server entry's <code>uid</code></li> <li>2. Create a new entry in Active Directory for the contractor</li> <li>3. Run <code>idsync resync</code> to establish a link between the new full-time employee. The <code>-a</code> option can be used to limit the scope of the <code>resync</code> to a single user</li> </ol> <p>Or if there is nothing important in the Directory Server entry:</p> <ol style="list-style-type: none"> <li>1. Delete the Directory Server entry for the contractor if there was one already</li> <li>2. Create a new entry in Active Directory</li> <li>3. Identity Synchronization for Windows will create the corresponding new user in Directory Server</li> </ol>
Yes	No	<ol style="list-style-type: none"> <li>1. Remove the <code>c-</code> prefix from the Active Directory entry's <code>samaccountname</code></li> <li>2. Identity Synchronization for Windows will interpret the change as a new user and will create the corresponding new user in Directory Server</li> </ol>
Yes	Yes	<ol style="list-style-type: none"> <li>1. Remove the <code>c-</code> prefix from the Directory Server entry's <code>uid</code></li> <li>2. Remove the <code>c-</code> prefix from the Active Directory entry's <code>uid</code>.</li> </ol> <p><b>Note:</b> If this entry was modified before the Directory Server entry, then the contractor would have two Directory Server accounts (the original one and a new one with a <code>uid</code> without the <code>c-</code> prefix)</p> <ol style="list-style-type: none"> <li>3. Run <code>idsync resync</code> to establish a link between the new full-time employee. The <code>-a</code> option can be used to limit the scope of the <code>resync</code> to a single user</li> </ol> <p>Or if there is nothing important in the Directory Server entry:</p> <ol style="list-style-type: none"> <li>1. Delete the Directory Server entry for the contractor if there was one already</li> <li>2. Remove the <code>-c</code> prefix from the Active Directory entry's <code>samaccountname</code></li> <li>3. Identity Synchronization for Windows will create the corresponding new user in Directory Server</li> </ol>



# Case Study: Deploying in a High-Availability Environment Over a Wide Area Network Using SSL

The case study provided in this chapter explains how Global Telco implemented Identity Synchronization for Windows in a four-way Multi-Master Replication (MMR) environment. This chapter also highlights important features of Identity Synchronization for Windows, such as:

- Integrating Identity Synchronization for Windows with Sun Java™ System Identity Manager to synchronize passwords
- Deploying Identity Synchronization for Windows in a high-availability (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

The information in this chapter is organized into the following sections:

- “Global Telco Deployment Information” on page 80
- “Installation and Configuration Overview” on page 84
- “Configuration Walkthrough” on page 89
- “Failover Installation” on page 97
- “Setting Up SSL” on page 101
- “Increasing Connector Worker Threads” on page 106

- “Aligning Primary and Failover Configurations” on page 108
- “Initial idsync resync Operation” on page 111
- “Periodic idsync resync Operations” on page 115
- “Configuring Identity Manager” on page 122
- “Understanding the Failover Process” on page 122
- “Initializing the Connector State” on page 124
- “Failover Installation Maintenance” on page 126
- “When to Failover” on page 126
- “Failing Over” on page 128

## Global Telco Deployment Information

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

They have 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 masters and four read-only replicas.

### Directory Server Setup

The Sun Java System Directory Server topology includes four masters and four read-only 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 the users are stored in the `ou=people,dc=gt,dc=example,dc=com` container with `uid` as the naming attribute.

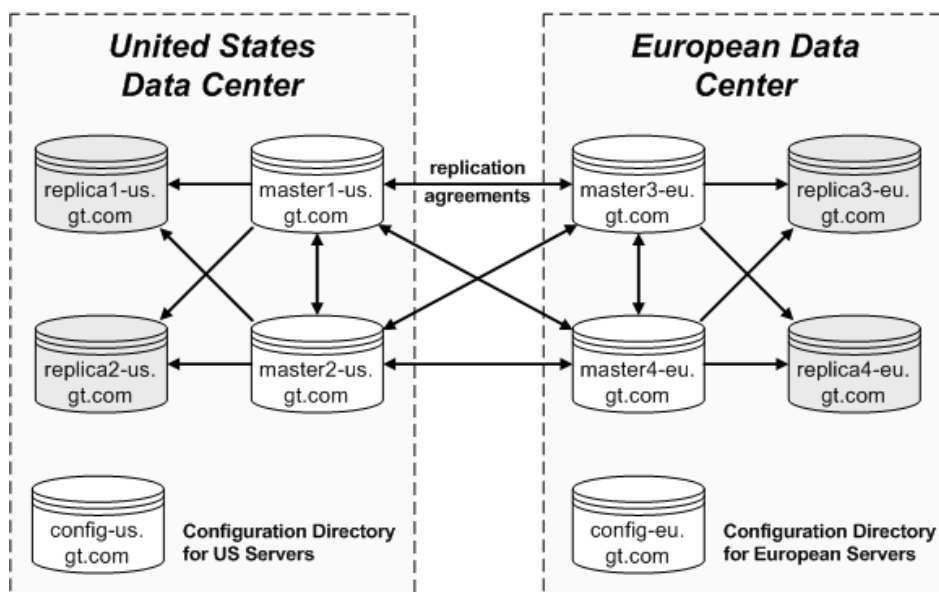
- Two masters and two read-only replicas are located in the United States (a separate configuration directory in the United States stores configuration information for these systems).
- Two masters and two read-only replicas are located in Europe (a separate configuration directory in Europe stores configuration information for these systems).



All four masters have replication agreements with each other, but the four read-only replicas only have replication agreements with two of the masters as shown in the following figure.

**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 master.

**Figure 3-1** Data Center Information for Directory Server



## Active Directory Information

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`).

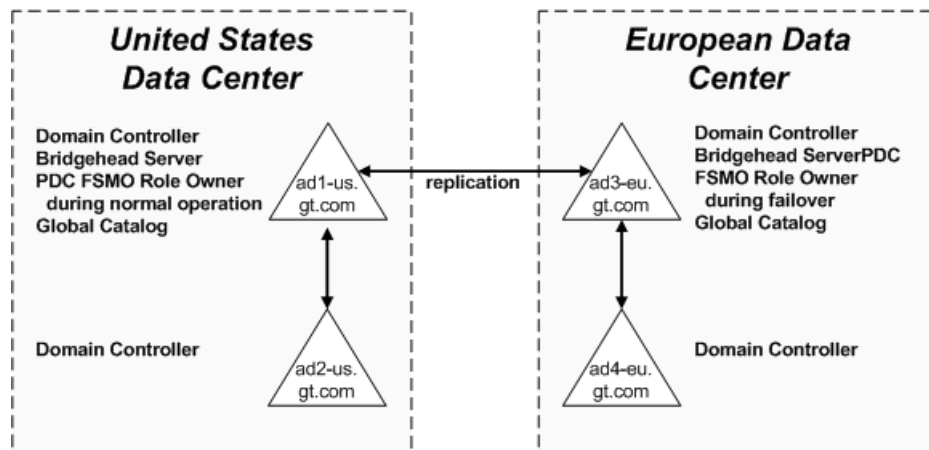
The Active Directory `samaccountname` attribute value matches the Directory Server `uid` attribute, and the Active Directory domain controller with the PDC FSMO role is located in the United States office.

---

**NOTE** Both `ad1-us.gt.com` and `ad3-eu.gt.com` are bridgehead servers which control replication between the two sites.

---

**Figure 3-2** Data Center Information for Active Directory



# Requirements

Global Telco wants to achieve the following:

- The users' passwords for Windows systems must be synchronized with their Directory Server passwords.
- The users must be able to change passwords using native mechanisms made in either environment, through the Change Password option in the Task Manager dialog on Windows systems, and 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.

---

<b>NOTE</b>	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, this is not advisable because it circumvents password policy and password history.
-------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

---

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

Identity Synchronization for Windows requires the users' Directory Server accounts to be explicitly linked to their Windows accounts. This 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 `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 for propagating native password changes made in Directory Server to all systems managed by Identity Manager must be added.

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, these two products can be coupled 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. Once 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 an alternate Directory Server master, 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 with the Identity Synchronization for Windows Core or Connector has a hardware failure, then Identity Synchronization for Windows will not synchronize users until it is re-installed 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.

## Installation and Configuration Overview

This section gives an overview of the installation and configuration steps for meeting Global Telco's requirements. For details on configuring Identity Manager to coexist with Identity Synchronization for Windows, see Appendix B.

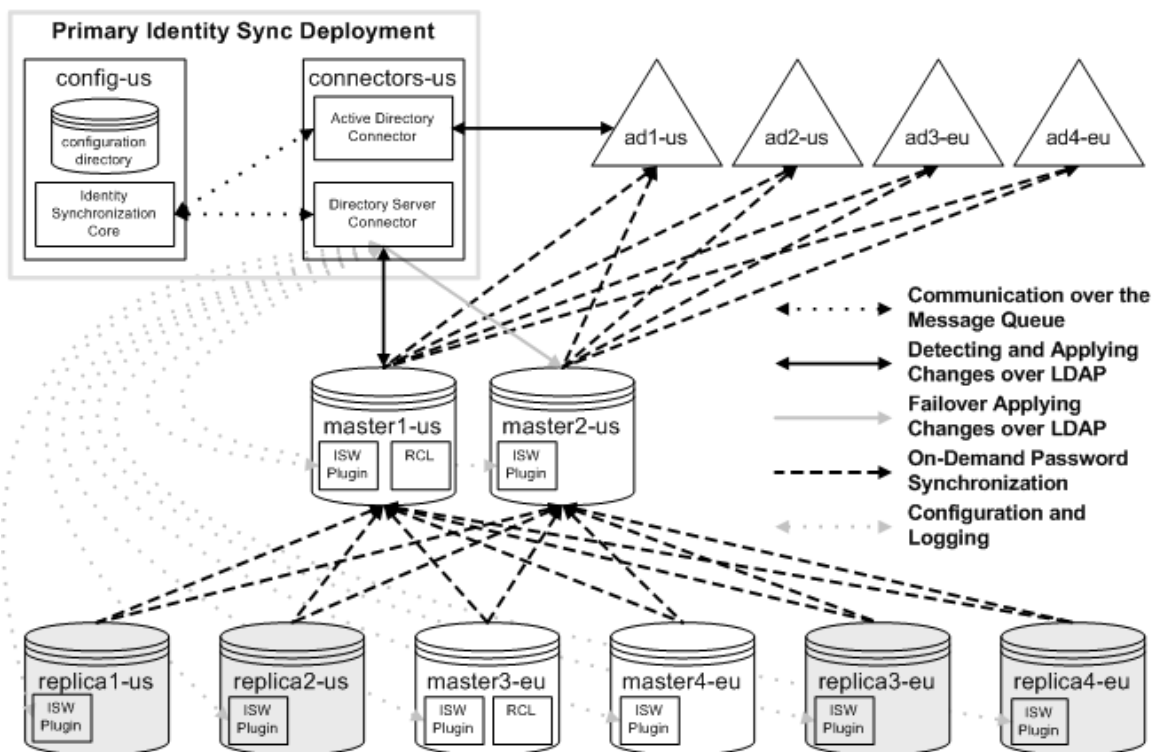
### Primary and Secondary Installations

To provide a high-availability solution, Identity Synchronization for Windows must be installed in two separate environments, 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 environment. 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 Directory Server masters. While it is available, it detects and propagates changes to `master1-us.gt.com`. If this machine is unavailable, it fails over to `master2-us.gt.com` to apply changes, but cannot detect further changes made at any master until `master1-us.gt.com` is available.

---

**NOTE**

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

Identity Synchronization for Windows' Directory Server Plugin must be installed on all eight Directory Server instances, four masters and four read-only replicas. When a directory server starts up, the Directory Server Plugin establishes a secure connection to the Directory Server Connector. Once the plugin is authenticated, the connector sends the configuration information, and the plugin 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 `dspswvalidate` attribute to `true` in the user's Directory Server entry. Because the user can log into any directory server, on-demand password synchronization can originate from any server.

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

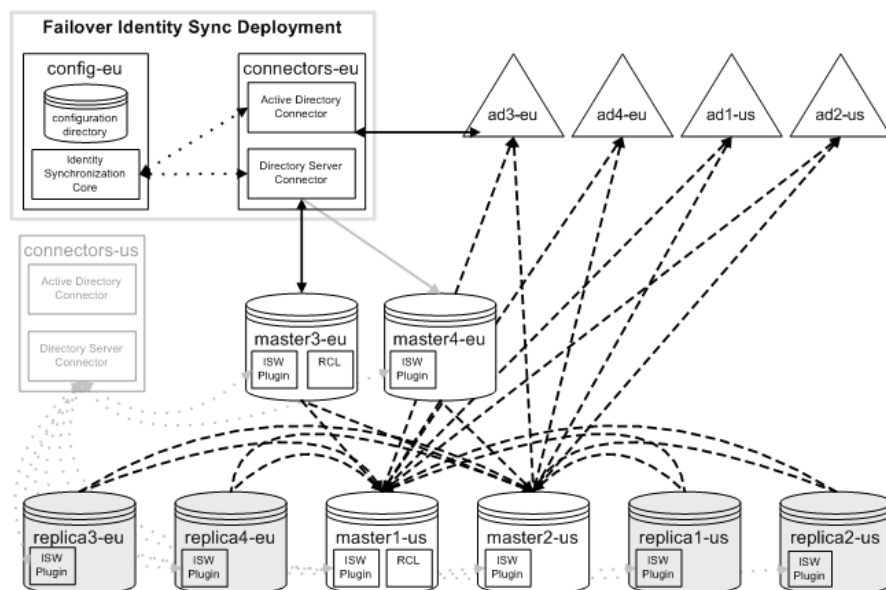
If the user logs into one of the other two masters or one of the read-only replicas, then on-demand password synchronization is done against `master1-us.gt.com` or `master2-us.gt.com`, and these masters in turn continue the on-demand password synchronization to one of the Active Directory domain controllers.

These two hops are necessary because:

- 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 masters, Identity Synchronization for Windows treats all Directory Server instances as read-only replicas.

After the Primary Installation is complete, the second Identity Synchronization for Windows installation is done on the two machines in Europe, `config-eu.gt.com` and `connectors-eu.gt.com` as shown in the figure below.

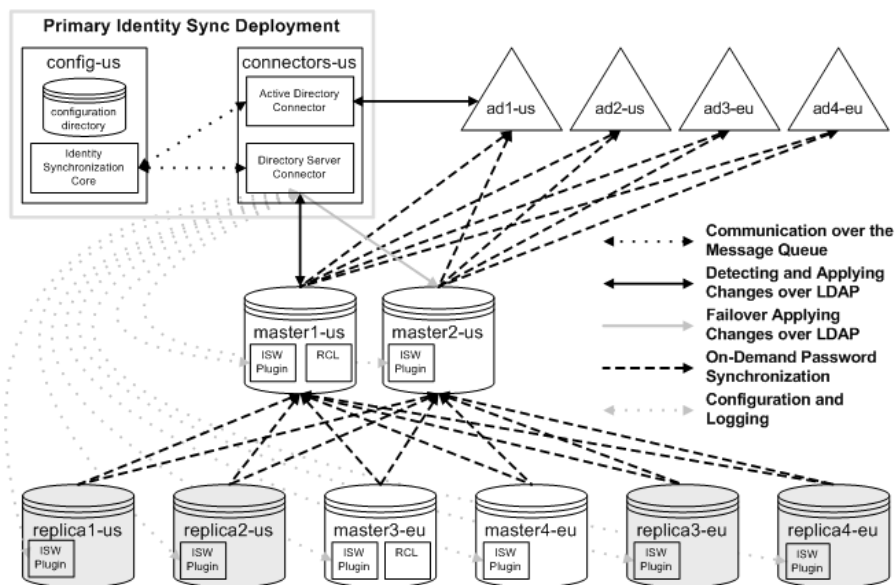
**Figure 3-4** Failover Installation while the Primary Installation is Active



The Identity Synchronization for Windows Directory Server Plugins have not been re-installed, so they still receive their configuration from the Directory Server Connector running on `connectors-us.gt.com`, while the 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 failover to it. To complete the failover process, the Identity Synchronization for Windows Plugin is reinstalled on every directory server. This 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 Plugins



**NOTE** Setting up the secondary installation significantly increases the amount of time required to deploy Identity Synchronization for Windows. However, this up front cost is offset by the ability to quickly failover 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, `idsync resync` command is executed periodically to establish links between newly provisioned users. An LDAP filter is passed to this command to resynchronize only the subset of users that were created since the command was last executed.

See “Periodic `idsync resync` Operations” on page 115 for more information.

## Large Deployment Considerations

Due to the large size of their 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.
- The Message Queue Broker's memory limits are increased (see the *Sun Java System Identity Synchronization for Windows Installation and Configuration 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 items is discussed in detail in the next section.

## Configuration Walkthrough

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

---

**NOTE** Only important steps are provided and any configuration instructions already discussed in the Example Bank case study have been omitted.

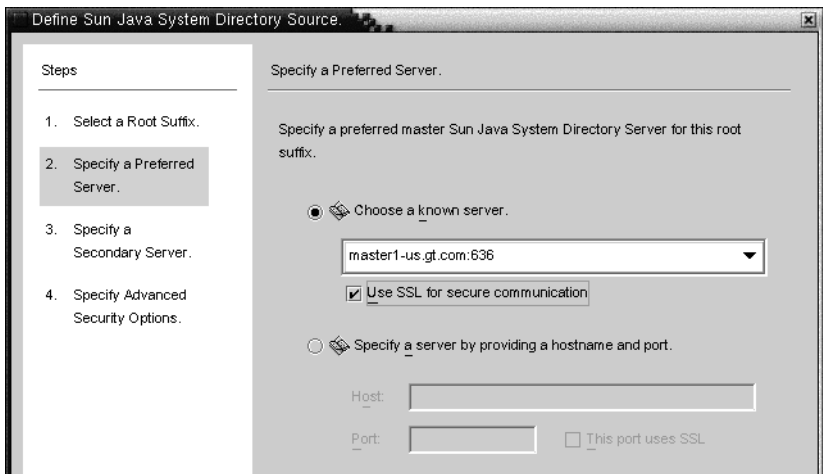
For detailed configuration instructions, see the *Sun Java System Identity Synchronization for Windows Installation and Configuration Guide*.

---

# Primary Installation

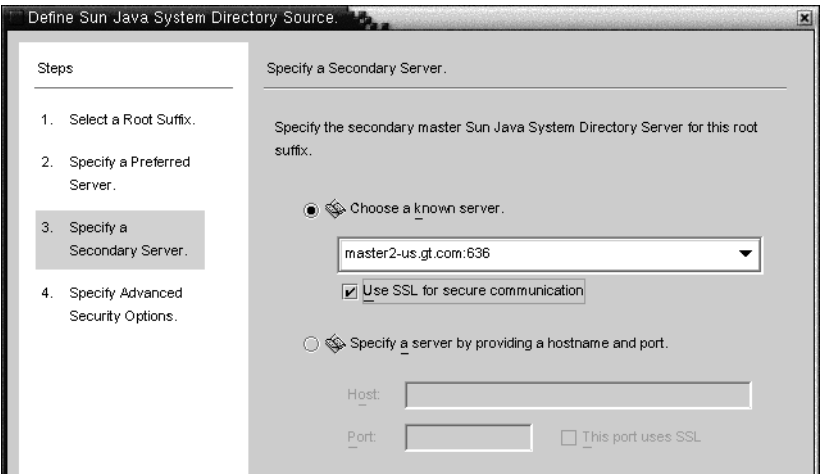
After the Core is installed on config-us.gt.com, the Identity Synchronization for Windows console is started. You configure the Directory Server source first.

**Figure 3-6** Configuring the Directory Server Source



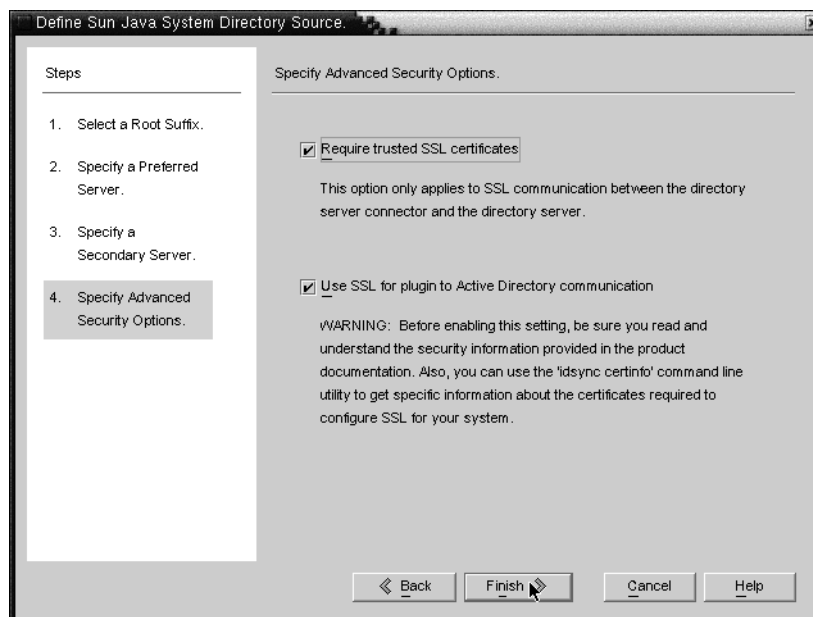
master1-us.gt.com is chosen as the preferred master. The connector communicates with the Directory Server source over SSL.

**Figure 3-7** Configuring the Directory Server Source Over SSL



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

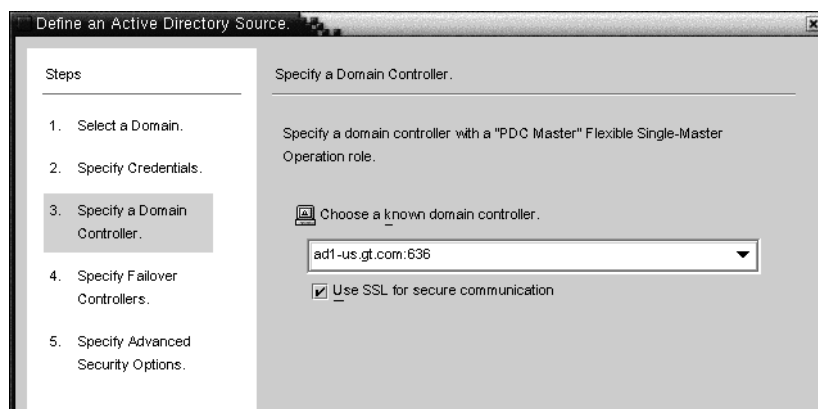
**Figure 3-8** Configuring Advanced Security Options for the Directory Server Source



Global Telco requires the strictest security possible, so the Directory Server Connector will require a trusted SSL certificate from the directory server, and the Identity Synchronization for Windows Directory Server Plugins will communicate over SSL to Active Directory. (The Identity Synchronization for Windows Plugins inherit the SSL configuration of the directory server. Therefore, if the Directory Server requires trusted certificates, the plugin can only communicate with Active Directory if it provides a trusted certificate). Enabling these enhanced security options implies additional installation step, outlined below.

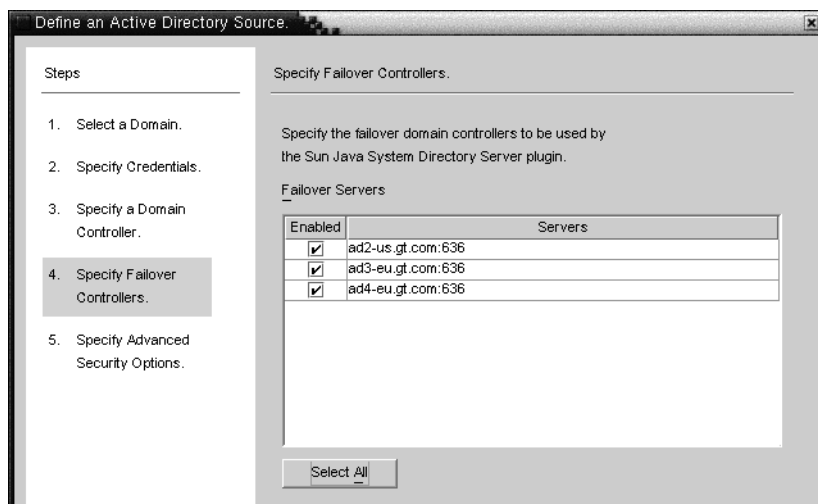
Next the information about the `gt.com` Active Directory domain is provided.

**Figure 3-9** Configuring the Active Directory Domain



`ad1-us.gt.com` is the PDC FSMO Role Owner, so it is selected as the domain with which the controller for the Active Directory Connector will communicate. The connector communicates over SSL.

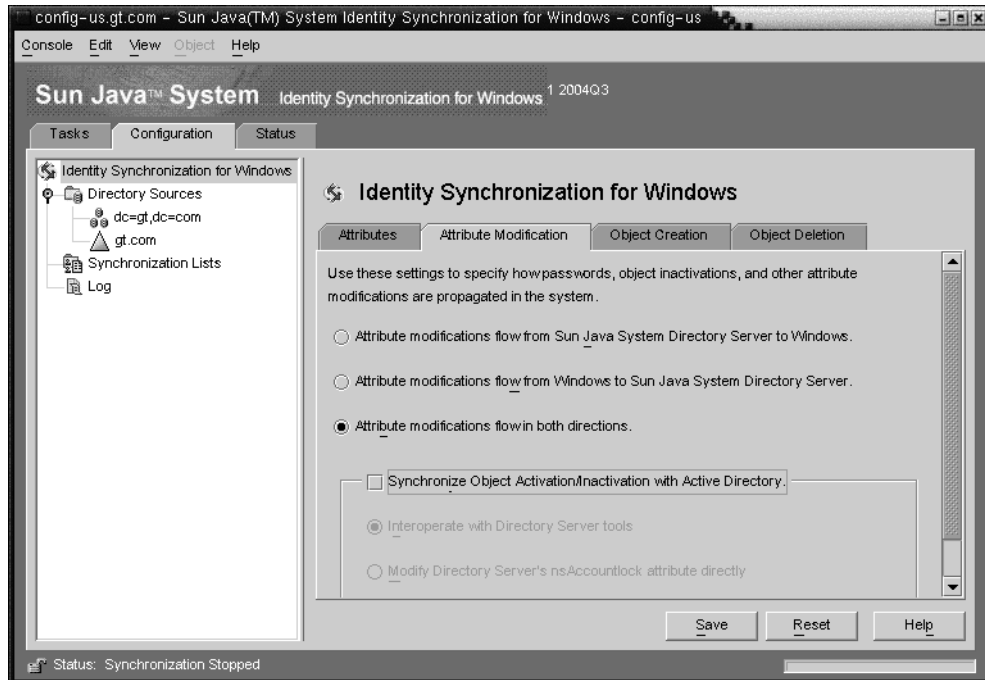
**Figure 3-10** Configuring Failover Active Directory Domain Controllers to Work over SSL



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

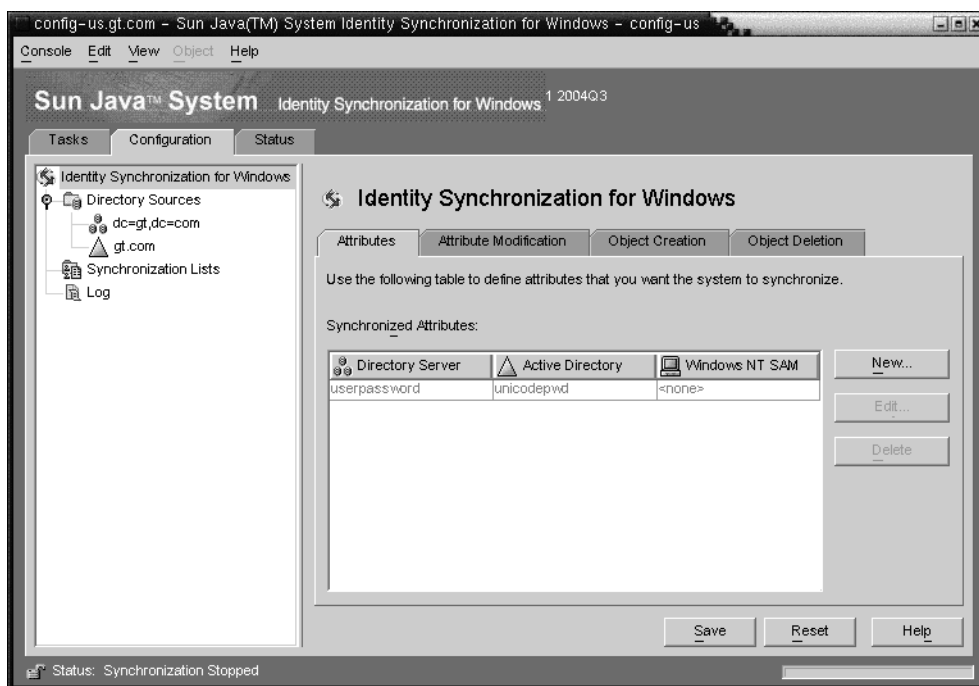
**Figure 3-11** Security Option to Enable for the Active Directory Connector

Global Telco requires the strictest security possible, so 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.

**Figure 3-12** Attribute Modification Flow Setting

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.

**Figure 3-13** Attribute Setting for Synchronization



Only passwords are synchronized. No additional attributes are synchronized.

A single SUL, `GT_USERS`, is created as shown in Figure 3-14.

**Figure 3-14** Synchronization User List Creation

The screenshot shows the 'Define a Synchronization User List' dialog box. On the left, a 'Steps' pane lists three steps: 1. Specify a Name., 2. Specify the Windows Criteria. (highlighted), and 3. Specify the Sun Java System Directory Server Criteria. The main area is titled 'Specify the Windows Criteria.' and contains the following fields:

- Directory Source:** A dropdown menu showing 'gt.com'.
- User Set Domain:** A section containing:
  - Base DN:** A text box with 'CN=Users,DC=gt,DC=com' and a 'Browse...' button.
  - Filter:** A text box with the LDAP filter '(!((cn=Administrator)(cn=Guest)(cn=TsInternetUser)(cn=krttgt)(cn=iswUser)))'.
- Creation Expression:** A text box with 'CN=Users,DC=gt,DC=com' and a note 'For example, cn=%cn%'.

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.

**Figure 3-15** Excluding User from the Synchronization Process

The screenshot shows the 'Define a Synchronization User List' dialog box. On the left, the 'Steps' pane lists three steps: 1. Specify a Name., 2. Specify the Windows Criteria., and 3. Specify the Sun Java System Directory Server Criteria. (highlighted). The main area is titled 'Specify the Sun Java System Directory Server Criteria.' and contains the following fields:

- Directory Source:** A dropdown menu showing 'dc=gt,dc=com'.
- User Set Domain:** A section containing:
  - Base DN:** A text box with 'ou=People,dc=gt,dc=com' and a 'Browse...' button.
  - Filter:** An empty text box.
- Creation Expression:** A text box with 'ou=People,dc=gt,dc=com' and a note 'For example, cn=%cn%'.

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 Plugin 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.

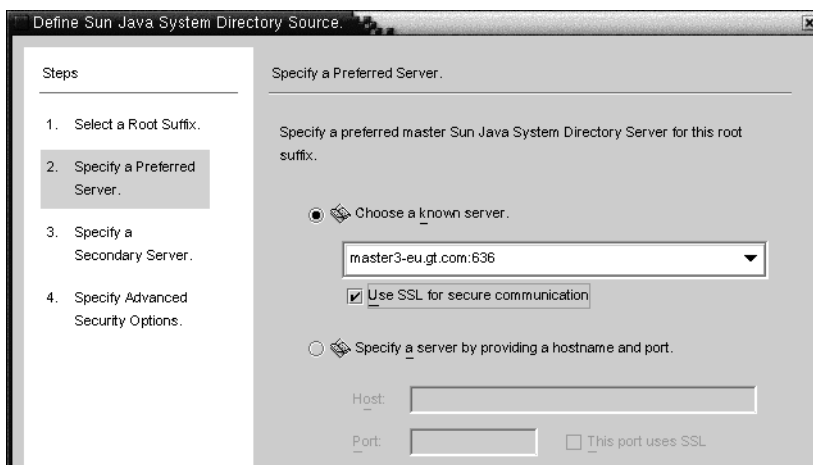
SUCCESS



## Failover Installation

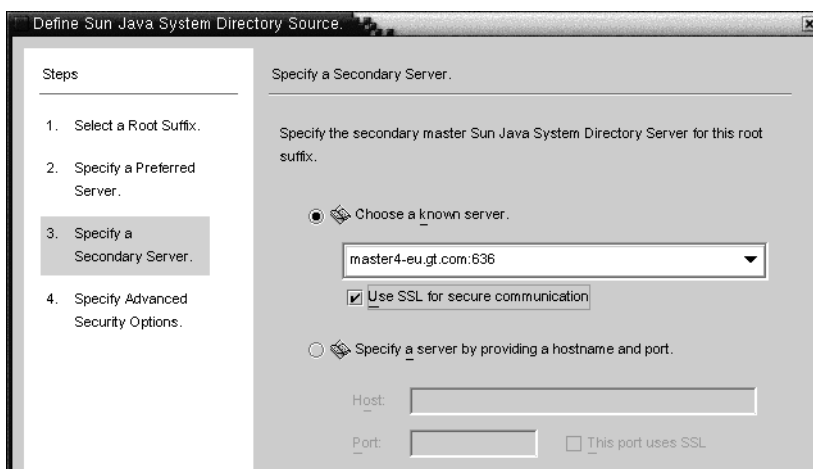
Once the primary installation is complete, the Identity Synchronization for Windows Core is installed on `config-eu.gt.com`, and the console is used to configure it.

**Figure 3-16** Configuring the Preferred Directory Server



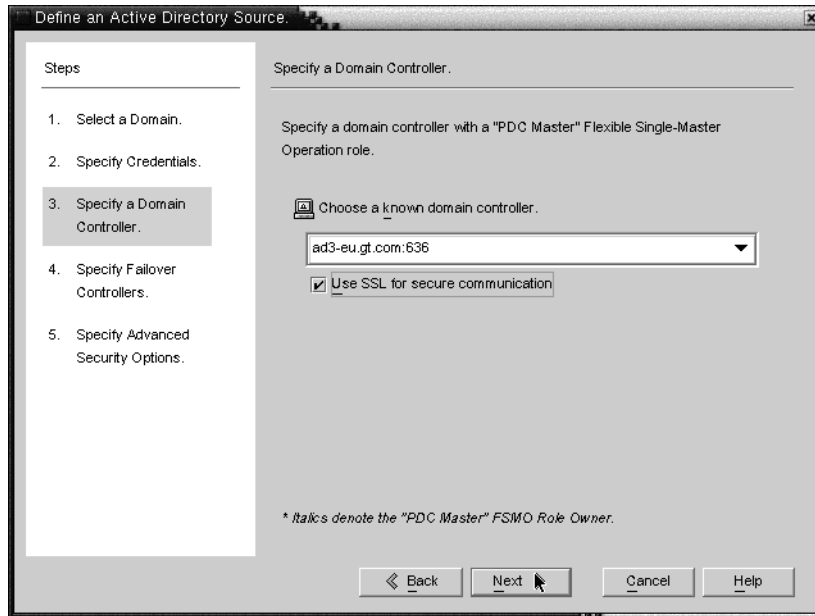
`master3-eu.gt.com` is the preferred Directory Server master in the failover installation.

**Figure 3-17** Configuring the Secondary Directory Server Master



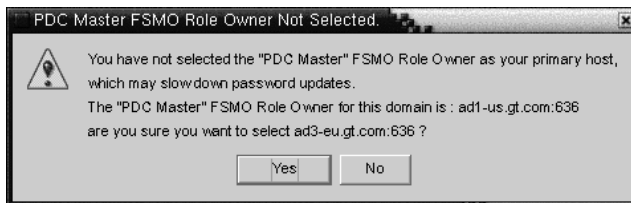
master4-eu.gt.com is the secondary directory server master in the failover installation.

**Figure 3-18** Configuring the Active Directory Domain Controller



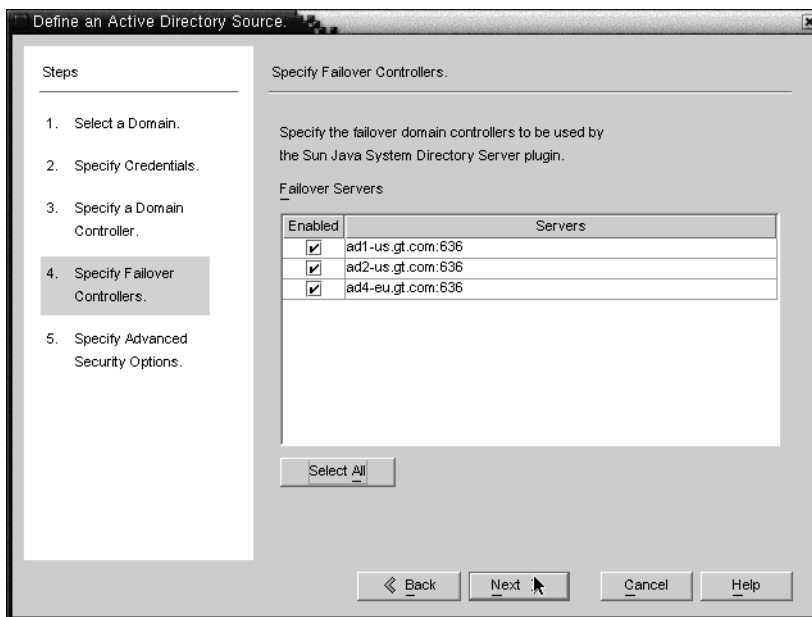
ad3-eu.gt.com is chosen as the domain controller with which the Active Directory Connector will communicate.

**Figure 3-19** Warning Message



This warning is displayed 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.

**Figure 3-20** Configuring Domain Controllers for Failover during On-Demand Synchronization



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

```
bash-2.05# /opt/SUNWisw/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 includes connections between:

- Directory Server Connector and the preferred and secondary directory server masters.
- Active Directory Connector and the Active Directory domain controller.
- Preferred and secondary directory server masters, and the Active Directory domain controllers for on-demand password synchronization.
- All other masters and read-only replicas and the preferred and secondary directory server masters 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 their Directory Server certificates and one to sign their Active Directory certificates. Both these CA certificates are added to the certificate database of each connector and each directory server instance.

---

<b>NOTE</b>	The <code>idsync certinfo</code> command displays the steps for configuring SSL for Identity Synchronization for Windows components, based on the current configuration. It 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 primary installation below. The output for the failover installation is identical except that the roles of the US and European machines is reversed.

```
bash-2.05# /opt/SUNWiwsw/bin/idsync certinfo -q <omitted password> -w <omitted password>
Connector: CNN100
  Installation Host: connectors-us
  Installation Path: /opt
  Certificate Database Location: /var/opt/SUNWiwsw/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/SUNWiw/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
```

Table 3-1 summarizes SSL communication between components in this installation, including trust requirements for the primary and failover installations.

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 only if the <b>Require trusted SSL certificates</b> option is enabled in the console.
	master2-us.gt.com	Primary	Only required if the <b>Require trusted SSL certificates</b> option is enabled in the console.
Active Directory Connector on connector-us.gt.com	ad1-us.gt.com	Primary	Only required if the <b>Require trusted SSL certificates</b> option is enabled in the console. The output of <b>idsync certinfo</b> 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 <b>Require trusted SSL certificates</b> option is enabled in the console.
	master4-eu.gt.com	Failover	Only required if the <b>Require trusted SSL certificates</b> option is enabled in the console.
Active Directory Connector on connector-eu.gt.com	ad3-eu.gt.com	Primary	Only required if the <b>Require trusted SSL certificates</b> option is enabled in the console. The output of <b>idsync certinfo</b> erroneously mentions that certificates for the other Active Directory domain controllers are required.



**Table 3-1** SSL Communication between Components

Component	Must Trust Certificates From	Required By	Comments
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. <b>idsync certinfo</b> 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. <b>idsync certinfo</b> does not mention this requirement.
	master4-eu.gt.com		
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. <b>idsync certinfo</b> 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. <b>idsync certinfo</b> does not mention this requirement
	master2-us.gt.com		
replica1-us.gt.com	master1-us.gt.com	Primary	Required for on-demand password synchronization. <b>idsync certinfo</b> 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. <b>idsync certinfo</b> does not mention this requirement.
replica4-eu.gt.com	master4-eu.gt.com		

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

---

**NOTE** See the *Sun Java System Identity Synchronization for Windows Installation and Configuration 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 Plugin is installed, therefore, it is recommended that you add the CA certificates to the Directory Servers' certificate databases before the Identity Synchronization for Windows Plugin is installed.

---

## Increasing Connector Worker Threads

By default each Directory Server and Active Directory connector uses four worker threads to apply changes. This value is increased in the Global Telco deployment to improve the connector performance, especially during `idsync resync` operations. The number of connector threads is stored in the configuration directory in the `pswNumOutboundConnectorThreads` attribute, present in the `pswSunDirectoryGlobals` and in the `pswActiveDirectoryGlobals` entries. 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=IdentityS ynchronization,ou=Services,dc=gt,dc=com
pswNumOutboundConnectorThreads: 4
```

The entry that must be modified is:

```
cn=136,ou=Sun,ou=Globals,cn=active[13],ou=GlobalConfig,ou=1.1,ou=IdentityS
ynchronization,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:

```
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 Core machine is restarted to notify the System Manager to pick up the new configuration.

---

<b>NOTE</b>	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, there are some generated configuration parameters that differ for 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 Plugin to encrypt passwords in the Retro-Change Log. This key is randomly generated when the configuration is first saved.

Both 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, the same encryption key cannot be used for both configurations, and therefore, some password changes might be lost during failover. The failover process includes re-installing the Identity Synchronization for Windows Plugins 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 is 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, we 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}k9AFSUGsYlNK038PvIB4lJzVNb0sQHh4JHJXFQ==
```

{SSHA}k9AFSUGsY1NK038PvIB4lJzVNb0sQHh4JHJXFQ== is the password that the Failover Directory Server Connector users to connect to the directory server. At this point, the Directory Server Connector for the primary installation will no longer be able to log into the directory, so we 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
```

Once this is complete, both Directory Server Connectors will be able to log into 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`. Once the connectors start and receive their configuration, they will open a connection to the directory. If there are any problems with the credentials, those will be 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 either Directory Server Connector is uninstalled and re-installed, 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, then 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 Operation

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

- 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 when a user logs into Directory Server for the first time.
- Primes the Active Directory Connector's object cache, so that changes to existing users are not falsely interpreted as newly created users.

Doing a full `idsync resync` operation is expensive in Identity Synchronization for Windows. For a deployment as large as gt.com (500K 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 `idsync resync` operation. To reduce this load, the following is done:

- Increase the Message Queue Broker's available memory to 1 GB as described in the installation guide.
- Change the Identity Synchronization for Windows system-wide log level from the default `FINE` to `INFO`, using the console.

---

<b>NOTE</b>	If you need to increase the log level to debug an issue, then it is recommended that you use the <code>-a</code> option during <code>idsync resync</code> operation to restrict the <code>idsync resync</code> operation to only the affected users.
-------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

---

- Divide the 500K users into five separate `idsync resync` operations using the `-a` option. This is described in more detail below.

These steps are also performed for the failover installation.

## Initial idsync resync Operation for Primary Installation

When you run `idsync resync` for the first time, the Active Directory users are divided 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 may not be English or even ASCII so resynchronizing, based on the 26 letters of the English alphabet, will skip users with non-English characters.
- The LDAP filters can overlap, however, a significant overlap will increase the total running time of the operation.
- Batch sizes are not too large; otherwise, the Message Queue Broker will reach 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`, then all entries must have a value for that attribute, to avoid entries being overlooked.
- If a user attribute used in the filter can change value during the operation, then 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 have been synchronized, then this user will be skipped. The script below 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, and might even speed the operation as 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, so these operations cannot be run in parallel.

The `-a <ldap-filter>` operation 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, so we 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 their `cn` attribute. `cn` is a mandatory attribute and 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`), then the user will be synchronized twice, but this introduces negligible overhead and does not cause 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` 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 will match all users 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/SUNWiw/samples/linkusers-simple-gt.cfg";
my $idsync = "/opt/SUNWiw/bin/idsync";
my $adHost = "adl-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`. This 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, they 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 it is a new user. If you run `idsync resync -k`, then 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

The `idsync resync` command should also be run for the failover installation. Because the links between entries were already established, `idsync resync` is only required to prime the object cache database for the Active Directory Connector. Therefore, it is run with the `-u` option. To avoid overloading Message Queue with log messages: the Message Queue Broker's maximum memory is increased, the Identity Synchronization for Windows log level is changed to `INFO`, and `idsync resync` is run over the entire user population in batches. The script above is modified slightly and run on the failover installation Core machine, `config-eu.gt.com`. The parameters at the top of the script are changed, and the `-i NEW_LINKED_USERS` option is replaced 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, then modifications to their passwords will not synchronize. Running a full `idsync resync` operation for 500 thousand users could take a few hours. Because synchronization of new changes is delayed during a

idsync resync operation, having Identity Synchronization for Windows off-line for long is unacceptable. Instead, Global Telco uses Active Directory's `usnCreated` attribute to synchronize only those users that have been created recently. They use the `resync-recent.pl` script given below, which is run every hour, to synchronize users that have been created in the last three days. This script is run over 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 adl-us.gt.com usnCreated -f
/var/opt/SUNWsw/samples/linkusers-simple-gt.cfg -i NEW_LINKED_USERS -b -w
- -q - < /var/opt/SUNWsw/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 only run on entries that were created recently, and 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`; it prevents passwords from appearing in the command line and being available to anyone on the system via commands such as `ps`. The configuration directory password and the configuration password are written to `/var/opt/SUNWsw/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 $IDSYNC = "/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 %usrHistory = ();
    if (!open(IN, "<$USN_FILE")) {
        print STDERR "Could not read history from $USN_FILE.\n";
        return %usrHistory;
    }

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

    return %usrHistory
}

```

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

- Add the `dspswuser` objectclass to the user entry.
- Set the `dspswuserlink` attribute to match the user's `objectguid` attribute from Active Directory.
- Set the `dspswvalidate` attribute to `true` to force on-demand password synchronization. This is optional.

## 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, then the Active Directory Connector will detect and propagate many changes that were already synchronized by the primary installation. This will significantly increase the load, and will place a heavier load on Directory Server during the failover scenario.

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

```
resync-recent.pl ad3-eu.gt.com usnCreated -u -b -w - -q - <
/var/opt/SUNWisw/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), then 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, then Identity Synchronization for Windows should at least report the changes that are lost.

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

- 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 can process all changes that occurred when it was not running, by 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, then the connector will detect only new changes in Directory Server.

This has an effect on the failover process. If synchronization is never started for the failover installation, then 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 Plugin 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). Once synchronization is started for the failover installation, the Directory Server Connector only processes one day's worth of changes in the directory server.

The *Sun Java System Directory Server Administration Guide* contains instructions on how to enable trimming of the retro changelog. Essentially, the `nsslapd-changelogmaxage` attribute on the `cn=Retro Changelog Plugin`, `cn=plugins,cn=config` entry is modified and Directory Server is restarted. Setting this value to `1d` trims entries that are older than one day. The value can be modified from the command line 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 it 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 checkpoint.

When failing over and synchronization is restarted for the failover installation, the Active Directory Connector will process all entries that were modified because it was last started. Even though this will be thousands of entries, the processing will be fast because the Active Directory's object cache database is up-to-date with most of the changes. Only the changes made since the last time that `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 Plugin must be re-installed on both `master3-eu.gt.com` and `master4-eu.gt.com` to point to the failover configuration. Once the plugin has been installed and the directory servers have been restarted, synchronization can be started. Verify that both connectors have entered the `SYNCING` state 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

```

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

1. **Directory Server Connector:** Check for the presence of the `/var/opt/SUNWsw/persist/ADP100/accessor.state` file. And 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`, and then replace CNN with ADP in the connector ID.)
2. **Active Directory Connector:** Check that the Active Directory Connector actually propagated the change. There should be an `INFO` message in the central log that includes the `usnchanged` value, 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)
```

Once you have verified that both the connectors have check-pointed their state, stop synchronization for the failover installation, and then re-install the Directory Server Plugins on `master3-eu.gt.com` and `master4-eu.gt.com` to point to the primary configuration.

## Failover Installation Maintenance

Once the connectors' state has been persisted, little maintenance is done on the failover installation. `idsync resync -u` should be run periodically as described in the section "Periodic `idsync resync` Operation for Failover Installation" on page 121, and any configuration changes made at the primary installation should also be made at the failover installation.

---

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

---

## When to Failover

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, then most users will be unaffected. Once 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

Plugin to Active Directory. If on-demand password synchronization fails, then the user will not be able to log into Directory Server. Therefore, Identity Synchronization for Windows provides more availability options for this area. The Directory Server Plugin can be configured to authenticate to any Active Directory domain controller, so even if all but one Active Directory domain controller is down, on-demand password synchronization will still succeed.

---

<b>NOTE</b>	The Directory Server Plugins 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 plugin, 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, then the plugin's cached configuration is lost, and on-demand synchronization at that Directory Server will fail until the Directory Server Connector is available.
-------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

---

Depending on the size of the deployment, the failover procedure might take anywhere from 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 Identity Synchronization for Windows Core machine. Failover is recommended only in situations where Identity Synchronization for Windows must be completely re-installed or a complete `idsync resync` operation must be run over a large population.

For example:

- Any machine where the Identity Synchronization for Windows Core or a Connector run has 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 master is corrupted and must be initialized from another master.

# Failing Over

The failover process, at a high-level, only involves the following steps:

- “Stopping Synchronization at the Primary Installation”
- “Starting Synchronization at the Failover Installation”
- “Re-installing the Directory Server Plugins”
- “Changing the PDC FSMO Role Owner”
- “Monitoring the Logs”

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



## Starting Synchronization at the Failover Installation

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

---

**NOTE** The Directory Server Connector will not enter the `SYNCING` state until the Directory Server Plugins are re-installed on the preferred and secondary masters (`master3-eu.gt.com` and `master4-eu.gt.com`).

Because the Active Directory Connector will need to process every entry in Active Directory that was modified since it was last started, it might take several minutes for it 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-installing the Directory Server Plugins

To complete the failover process, the Directory Server Plugin is reinstalled on each Directory Server. This ensures:

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

The plugins must be re-installed in this order:

1. Failover installation's preferred master.
2. Failover installation's secondary master.
3. All other masters.
4. All read-only replicas.

---

**NOTE** The order in which the Directory Server Plugins are re-installed is important. If they are installed in the wrong order, on-demand synchronization requests could loop between two masters, tying up all Directory Server connections.

---

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

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

1. Install the Directory Server Plugins for the Failover configuration.
2. Export the plugins' configuration for each master from the `cn=pswsync,cn=plugins,cn=config` tree. This will include two entries.
3. Re-install the Directory Server Plugins for the Primary configuration.

To failover:

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

## Changing the PDC FSMO Role Owner

This step 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, then 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."
```

This will occur for each password update in the Retro changelog that was made before the Directory Server Plugin was re-installed because the Primary Directory Server Plugin was configured to use a different encryption key from the failover Directory Server Plugin. Many of these password updates were likely synchronized by the primary installation before it went off line, but those that occurred after the primary installation went offline cannot be recovered. These users 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 Identity Synchronization for Windows and Pluggable Authentication Methods (PAM) so an LDAP store can provide synchronization capabilities between Solaris and Windows. The information is organized into the following sections:

- “Overview” on page 134
- “Configuring PAM and Identity Synchronization for Windows” on page 136
- “Configuring Systems to Prevent Eavesdropping” on page 158
- “Introducing Windows NT into the System” on page 158
- “Example /etc/pam.conf File” on page 160

---

<b>NOTE</b>	In this appendix, <i>Windows</i> refers to Windows environments using Active Directory for authentication. Windows NT environments may impose different approaches.
-------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------

---

# Overview

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

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, enable user information (including passwords) created or modified on a Solaris host to flow to its Windows counterpart (and vice versa) so either system can act on the information.

- Use PAM to authenticate Solaris and to manage passwords against an LDAP store
- Enable users to change their own passwords (if doing so does not contradict policy)

In addition, you can configure your environment to ensure that passwords are never sent over a medium that permits eavesdropping.

Solaris' implementation of PAM has long-offered the ability to use an LDAP store; however with the advent of intrinsic PAM modules in Solaris 9 you can now use a product such as Identity Synchronization for Windows.

---

<b>NOTE</b>	You can patch Solaris 8 to support this functionality using Patch number 108993 for Sparc or Patch number 108994 for x86.
-------------	---------------------------------------------------------------------------------------------------------------------------

---

While some Solaris 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) against 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 then `PAM_UNIX` makes its authentication decision accordingly.

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

Specifically, to initiate the synchronization process, Identity Synchronization for Windows requires all authentication systems to bind to the LDAP store. Furthermore, the binding mechanism you use must be a form that presents the user's password in a non-obfuscated manner, such as a simple bind, which precludes the use of 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 pre-formed, "user-centric" DN and a user-provided password when authenticating. This action in particular allows Identity Synchronization for Windows to maintain the synchronization of an entry. Consequently, you will use this `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 environments using Active Directory for authentication. Windows NT environments may require different approaches.

---

Use the following steps to configure Identity Synchronization for Windows and PAM for environments in which passwords can change on Windows:

1. Configure an LDAP repository for PAM (page 136)
2. Install and configure Identity Synchronization for Windows to synchronize the LDAP repository and Windows (page 139)
3. Populate the LDAP repository with user data (page 140)
4. Configure a Solaris host to use the LDAP repository (page 144)
5. Establish password authentication through the LDAP repository and enable users to change their own passwords (page 150)
6. Verify that user information (including passwords) flows between environments (page 152)

The rest of this section provides detailed information about each step and uses examples to illustrate the process.

## Step 1: Configure an LDAP Repository for PAM

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

- The LDAP store is a Sun One Directory Server 5.2 system that is hosted in a Solaris 8 environment.
- 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.

---

**NOTE** Before you begin, consult the *Sun Java System Identity Synchronization for Windows 1 2004Q3 Installation and Configuration Guide* to verify that you are using a supported directory server.

---

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

1. Configure the LDAP store using the Solaris `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 (Figure A-1):

**Figure A-1** Summary of Configuration Screen

```

Summary of Configuration
1 Domain to serve           : example.com
2 Base DN to setup         : dc=pam,dc=example,dc=com
3 Profile name to create    : pamconfig
4 Default Server List       : 192.168.220.219
5 Preferred Server List     :
6 Default Search Scope      : one
7 Credential Level          : proxy
8 Authentication Method      : simple
9 Enable Follow Referrals    : FALSE
10 iDS Time Limit           :
11 iDS Size Limit           :
12 Enable crypt password storage : TRUE
13 Service Auth Method pam_ldap : pam_ldap:simple
14 Service Auth Method keysevr : keysevr:simple
15 Service Auth Method passwd-cmd: passwd-cmd:simple
16 Search Time Limit        : 30
17 Profile Time to Live      : 43200
18 Bind Limit               : 10
19 Service Search Descriptors Menu

```

2. Evaluate the following key parameters' values:

- o Domain to serve
- o Base DN to setup
- o Profile name to create
- o 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 scenario. If you are working in a test environment where you can change DNS entries and set machine IP addresses to arbitrary values, you could use the names and addresses provided in this appendix.

3. After `idsconfig` stores the generated configuration, the `idsconfig` tool will direct you to create virtual list view (VLV) indexes.

---

<b>NOTE</b>	VLV indexes (also called <i>browsing indexes</i> ) enable PAM to quickly search for groups, users, and so forth. Refer to the following website for information about creating VLV indexes:
-------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

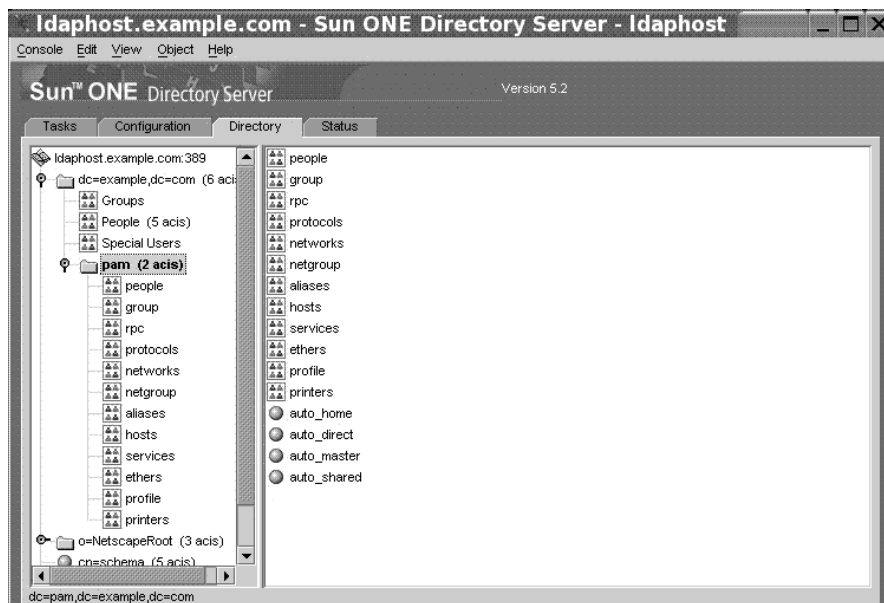
<http://docs.sun.com/source/816-6699-10/scripts.html#15172>

---

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.

Figure A-2 shows the resulting topology, as seen from the Sun One Directory Server console:

**Figure A-2** Resulting Topology



When you are finished configuring the LDAP repository for PAM, continue to “Step 3: Populating the LDAP Repository” on page 140.

## Step 2: Configuring Identity Synchronization for Windows

Now you can begin the process of bridging the LDAP store with the Windows' authentication system. You accomplish this process by installing and configuring Identity Synchronization for Windows against the following two systems:

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

You can install Identity Synchronization for Windows on the Solaris 8 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.

---

**NOTE** Instructions for installing and configuring Identity Synchronization for Windows are provided in the *Sun Java System Identity Synchronization for Windows 1 2004Q3 Installation and Configuration Guide* and the *Sun Java System Identity Synchronization for Windows 1 2004Q3 Release Notes*.

---

When you finish configuring Identity Synchronization for Windows, continue to “Step 3: Populating the LDAP Repository.”

## Step 3: Populating the LDAP Repository

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

Assume you want to 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 want to use an `ou=people` “container” that is subordinate to the Base DN you provided to `idsconfig` in Step 2 on page 138. You may have to make contextual changes to the Base DN you are going to use.

Use the following steps to populate the LDAP repository:

1. From the Directory tab in the Sun ONE Directory Server console, right-click on the people node (located just below the highlighted pam (2 acis) entry) and when the pop-up menu is displayed, select New -> User.
2. When the Sun One Directory Server's Create New User dialog box opens, enter the appropriate information in the text fields provided (for example, see Figure A-3).

**Figure A-3** Creating a New User

**Create New User**

Phone:  
Fax:

User  
Languages  
NT User  
Posix User  
Account

\* First Name: George  
\* Last Name: Washington  
\* Common Name(s): George Washington  
User ID: gwashing  
Password: \*\*\*\*\*  
Confirm Password: \*\*\*\*\*  
E-Mail: gwashington@example.com (e.g., user@company.com)  
Phone: 555 555-1212  
Fax:  
\* Indicates a required field

3. Select Posix User from the list in the left pane and the right pane changes so you can specify Posix User attributes.
4. Enable the Enable Posix User Attributes button and then enter the appropriate information in the text boxes provided (see Figure A-4).

**Figure A-4** Enabling Posix User Attributes

**Create New User**

Phone:  
Fax:

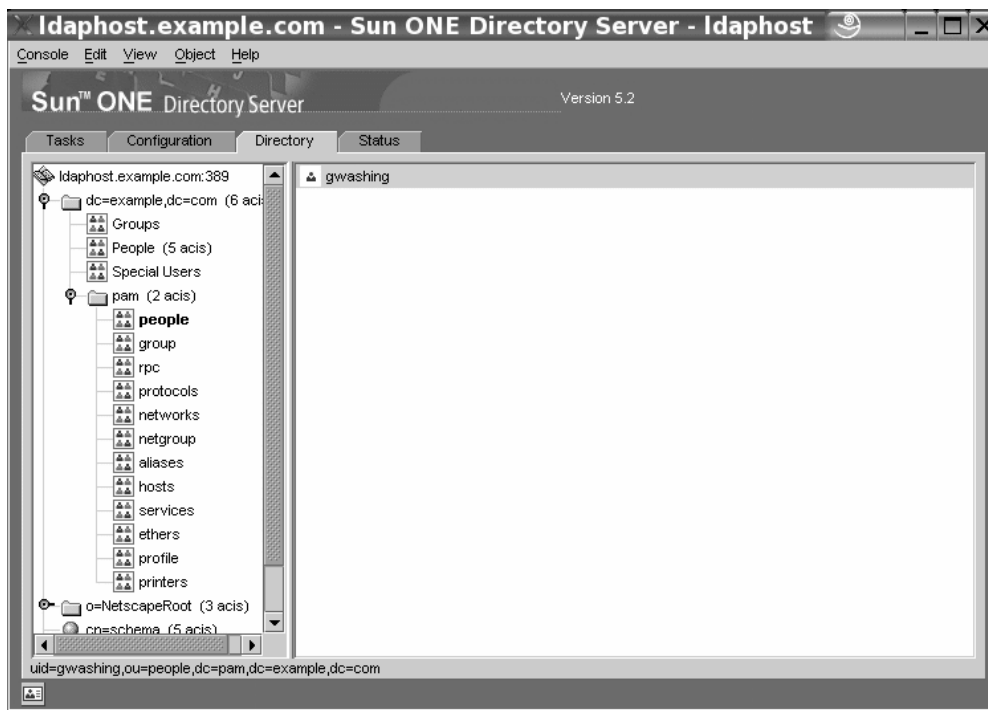
User  
Languages  
NT User  
Posix User  
Account

☒ Enable Posix User Attributes  
\* UID Number: 1000  
\* GID Number: 10  
\* Home Directory: /pres/home/gwashington  
Login Shell: /usr/bin/bash  
Gecos: George Washington  
\* Indicates a required field

5. When you are finished, click OK to save your changes and close the Create New User dialog box.

6. Verify that the new user (George Washington) is displayed in the console (Figure A-5).

**Figure A-5** Sun ONE Directory Server Console



7. You must edit the `gwashing` entry to add the `shadowaccount` objectclass.
  - a. Right-click on `gwashing` and when the pop-up menu is displayed, select the Edit with Generic Editor option.
  - b. In the Generic Editor dialog box, complete the appropriate text fields (for example, see Figure A-6).

**Figure A-6** Adding the shadowaccount Objectclass

Generic Editor - uid=gwashing,ou=people,dc=pam,dc=example,dc=com

createtimestamp	20040823194758Z
creatorsname	uid=admin,ou=administrators,ou=topologymanagement,o=netscaperoot
Email address	gwashington@example.com
entrydn	uid=gwashing,ou=people,dc=pam,dc=example,dc=com
entryid	29
First name	George
Full name	George Washington
gecos	George Washington
gidnumber	10
hassubordinates	FALSE
homedirectory	/pres/home/gwashington
Last name	Washington
loginshell	/usr/bin/bash
modifiersname	uid=admin,ou=administrators,ou=topologymanagement,o=netscaperoot
modifytimestamp	20040823194758Z
nsuniqueid	4eb93381-f53d11d8-8001a611-1ab2a823
numsubordinates	0
Object class	top person organizationalPerson inetorgperson posixAccount shadowaccount
parentid	11
Password	*****
subschemasubentry	cn=schema
Telephone number	555 555-1212
uidnumber	1000
User ID	gwashing

dn: uid=gwashing,ou=people,dc=pam,dc=example,dc=com

View

☐ Show Attribute Names

☒ Show Attribute Description

☒ Show only Attributes with Values

☒ Show DN

Refresh

Edit

Add Value

Delete Value

Add Attribute

Delete Attribute

Naming Attribute: uid Change...

OK Cancel Help

- When you are finished, click OK to save your changes and close the dialog box.

PAM clients can now authenticate against (and change the password for) this entry. Continue to “Step 4: Configuring a Solaris Host to Use PAM”.

## Step 4: Configuring a Solaris Host to Use PAM

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

To configure a Solaris host and create a PAM client you must

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

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

### Installing and Configuring a Solaris Test System

Install and configure a test Solaris system as an independent, standalone machine.

---

<b>NOTE</b>	To simplify this example, consider configuring a machine that is devoid of any naming service directives (such as NIS or NIS+).  Also, consider using a Solaris 9 x86 4/04 system, which contains several beneficial patches created for PAM and its associated subsystems.
-------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

---

### Configuring the Client Machine

---

<b>NOTE</b>	The following example instructions assume that you installed and configured the Solaris host as described in the previous section.
-------------	------------------------------------------------------------------------------------------------------------------------------------

---

You must configure a PAM client machine 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 `ldapclient` command, which stores the client's configuration information on the local host.

---

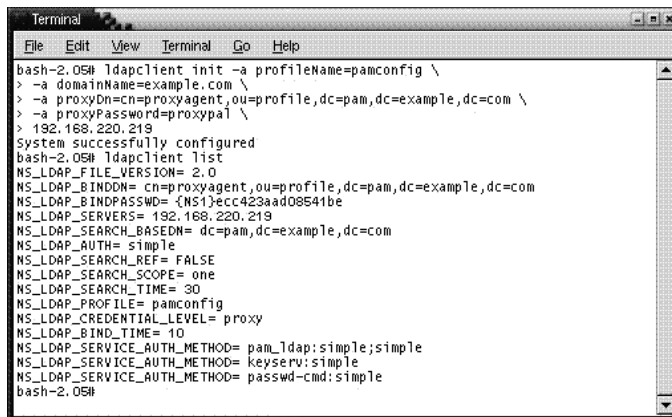
<b>NOTE</b>	Be sure to make a back-up copy of the <code>/etc/nsswitch.conf</code> file before you run the <code>ldapclient</code> command. Running <code>ldapclient</code> has several side effects — which includes completely replacing the system's <code>/etc/nsswitch.conf</code> file with a copy of the content in <code>/etc/nsswitch.ldap</code> .
-------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

---



Figure A-7 illustrates an example `ldapclient` command:

**Figure A-7** 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#

```

#### NOTE

- You should use an IP address for this configuration instead of a DNS name, because a DNS is not always available when the PAM system needs it.
- It is also important to use a proxy credential set to prevent anonymous authenticators from manipulating data in the LDAP store.

The system provides a set of proxy credentials 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 with a copy of the contents found in `/etc/nsswitch.ldap` (which is why you were advised to back-up the `/etc/nsswitch.conf` file). Consequently, most (or all) of the directives found in `/etc/nsswitch.conf` will include the LDAP directive (which means 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 may not be populated with the requisite host information needed to supplant DNS, the `/etc/nsswitch.conf` file is adjusted (which is the only change made to the post `ldapclient` command version of the `/etc/nsswitch.conf` file in this example).

You should edit the host's declaration to read as follows:

```
hosts: files ldap dns
```

Instead of the following reconfigured value (using `ldapclient`):

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

It is possible that this adjustment will not address your environment's needs correctly if you are running your DNS from the LDAP store. Consequently, be sure to apply this change only 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 accordingly.

## Specifying Rules for Authentication and Password Management

---

<b>NOTE</b>	The example instructions provided in this section assume that you installed and configured the Solaris host as described in the "Installing and Configuring a Solaris Test System" section.
-------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

---

The next step to perform when you are configuring a Solaris host to use PAM is to change the `/etc/pam.conf` file to incorporate the new rules you want it to use for authentication and password management. (See page 158 for an example `/etc/pam.conf` file.)

---

**CAUTION** Before making any changes to the `/etc/pam.conf` file, be sure you make a back-up copy of the original `/etc/pam.conf` file. Changes made to the `/etc/nsswitch.conf` and the `/etc/pam.conf` files can render your PAM client host inaccessible, which usually means that your configuration is set to deny everyone's (*including root*) authentication access to the machine.

To recover from this situation:

1. Leave a terminal/command window open on the machine and use this terminal session to adjust the `pam.conf` file for correctness.
2. In a new terminal/command window, try connecting to the localhost using the `rsh` or `ssh` command and then try logging in.
  - If your log-in fails to authenticate, you can still correct the problem using the open terminal/command window from Step 1.
  - If you “confuse” your Solaris system beyond the point of no return (assuming you are still logged in) your only viable option (short of reinstalling) is to restore the `/etc/nsswitch.conf` and `/etc/pam.conf` files back to their original state.

Using the Solaris `sys-unconfig` command may not restore your system because this command does not affect the `/etc/nsswitch.conf` and `/etc/pam.conf` files.

3. Repeat this trial and adjust procedure until you achieve the acceptable/expected system behavior.
- 

The changes you must make to `/etc/pam.conf` are trivial, but they must be explained. Essentially these changes fall into one of two categories:

- “Authentication” on page 148
- “Password Management” on page 149

## Authentication

For purposes of authentication, you must edit the 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 those entries to accept a binding directive and to pass the `server_policy` parameter to the `PAM_UNIX_AUTH` module.

Figure A-8 shows a diff between the original `/etc/pam.conf` file and the edited file.

**Figure A-8** Edited `/etc/pam.conf` 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_unix_auth.so.1 server_policy
30c31,32
< rlogin      auth required      pam_unix_auth.so.1
---
> rlogin      auth binding        pam_unix_auth.so.1 server_policy
> rlogin      auth required      pam_unix_auth.so.1 server_policy
36c38,39
< rsh auth required          pam_unix_auth.so.1
---
> rsh auth binding          pam_unix_auth.so.1 server_policy
> rsh auth required          pam_unix_auth.so.1 server_policy
42d44
< ppp auth required          pam_unix_auth.so.1
---
> ppp auth binding          pam_unix_auth.so.1 server_policy
> ppp auth required          pam_unix_auth.so.1 server_policy
50c54,55
< other auth required          pam_unix_auth.so.1
---
> other auth binding          pam_unix_auth.so.1 server_policy
> other auth required          pam_unix_auth.so.1 server_policy
54c59,60
< passwd      auth required      pam_unix_auth.so.1
---
> passwd      auth binding        pam_unix_auth.so.1 server_policy
> passwd      auth required      pam_unix_auth.so.1 server_policy
79c85
< other password required      pam_unix_auth.so.1
---
> other password required      pam_unix_auth.so.1 server_policy
bash-2.05#

```

2. Edit the file to add a new rule after the altered rule line.  
(Because the `/etc/pam.conf` file is processed from the top down, the line's order is important here.)

The new rule requires the service to include `PAM_LDAP` when deciding 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 local user log-ons. A *local user* is a user who is permitted by `/etc/nsswitch.conf` directives to examine files (such as the `root` account) *and* is enumerated in the `/etc/passwd` file. Local users are not necessarily stored in the LDAP store.

You might argue that allowing the `root` user to be listed in the LDAP store would simplify management of an important user account that spans the topology; however, you could make an equally powerful case 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, it is important to configure PAM in such a way so that PAM does not access the LDAP back-end store if the user has been described in the local files. You can address this situation by specifying the `server_policy` parameter for the `PAM_UNIX_AUTH` module in the given `/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. Of course, if the module cannot find a user either locally or in the LDAP store, then the system will provide an appropriate error message.

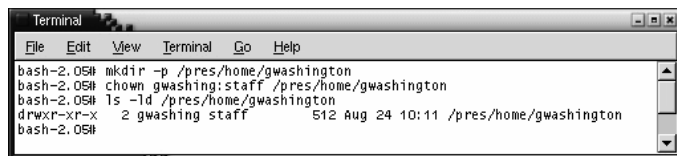
When you have finished configuring the Solaris host, continue to “Step 5: Verifying that PAM is Interoperating with the LDAP Store.”

## Step 5: Verifying that PAM is Interoperating with the LDAP Store

You are now ready to test whether the newly configured Solaris host can operate as a PAM client. However, before trying to log in as the example user, *George Washington*, you need to “cheat” just a bit.

If you review the screenshots of George’s user entry, note that George’s default home directory is `/pres/home/gwashington`. This directory does not yet exist on your test host nor have you configured the `auto_home` system on which to mount that file system automatically. Consequently, you can create the directory manually to avoid any unwanted complications (see Figure A-9).

**Figure A-9** Creating a Directory

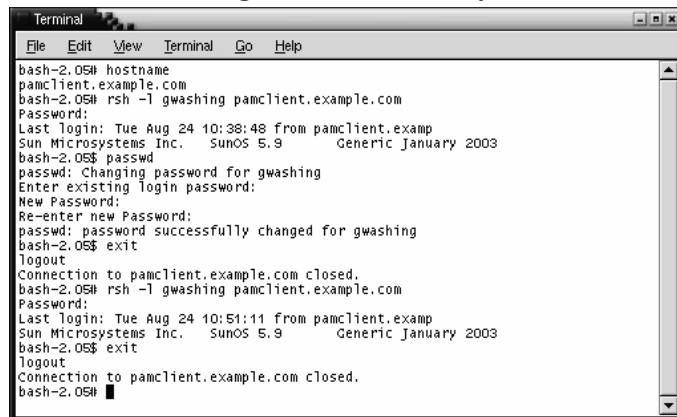


```

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#
  
```

You should be able to see the PAM system in action immediately (because `gwashing` is both understood and displayed). Figure A-10 shows that the PAM LDAP client system you configured can authenticate as `gwashing`. In addition, this figure demonstrates that a password change can be accomplished and that the new password will be accepted on a subsequent authentication.

**Figure A-10** Configured PAM LDAP System

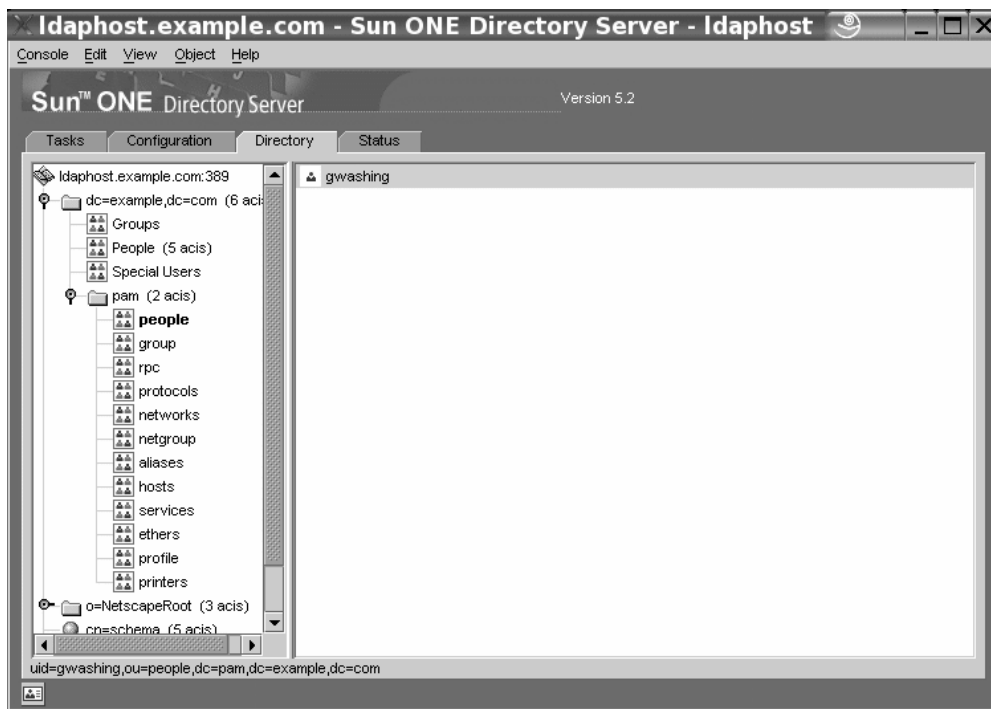


```

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#
  
```

If you check the LDAP store log (specifically looking for non-search operations) you should see an audit of the LDAP operations done in support of the preceding log-in and password-change test (for example, see Figure A-11).

**Figure A-11** Auditing LDAP Operations



## Step 6: Demonstrating that User Changes are Flowing to the Reciprocal Environment

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

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

- **Case 1** (page 152): Both Windows and the LDAP store contain current information
- **Case 2** (page 152): Windows is current but the LDAP store is outdated or *stale*
- **Case 3** (page 153): Windows is stale but the LDAP store is current
- **Case 4** (page 153): Both Windows and the LDAP store are stale

### Case 1

Case 1 is trivial — there is no supplementary work to do on either system because the entries are current.

### Case 2

In Case 2, the LDAP store is in a “holding pattern.” Specifically, the system is waiting for an opportunity to capture the current, correct password for the LDAP store. In this case, a properly configured Identity Synchronization for Windows system has marked the corresponding entry on the LDAP store's side of the equation as outdated or *stale*.

When an Identity Synchronization for Windows-enhanced Directory Server system gets a bind against 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 may eventually hash the value as part of its password policy) and clears the stale status.

This Identity Synchronization for Windows 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 against the LDAP store.



### Case 3

Case 3 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 pre-existing entries in the LDAP store presumably have one-way, hashed passwords. Consequently, when Identity Synchronization for Windows eventually 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 from the user the first time the user logs in.

If the potential of having multiple users sign on to your Windows environment without an effective password is not acceptable, you can address this issue by setting 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 either 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 log on. This policy facilitates the transformation of an end user's password from an overtly known value back into a covertly known value.

- When an end user changes their password on the LDAP store side and the Identity Synchronization for Windows system 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 just needs enough time to process the information and then the entry will be synchronized.

### Case 4

This case is a contradiction. If both systems are stale, then how can you ascertain which system contains the authoritative information? Human intervention will be required in this case, and your decision as to where to place the true state of an entry will then put the entry into one of the three, previously mentioned cases.

NOTE

This situation can degenerate into a very tedious process as you may be required to examine every user entry

For example, moving the entry from your LDAP store creates Case 3. If you created an new user called *George Washington* on a Solaris PAM machine, and then use an `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 follows:

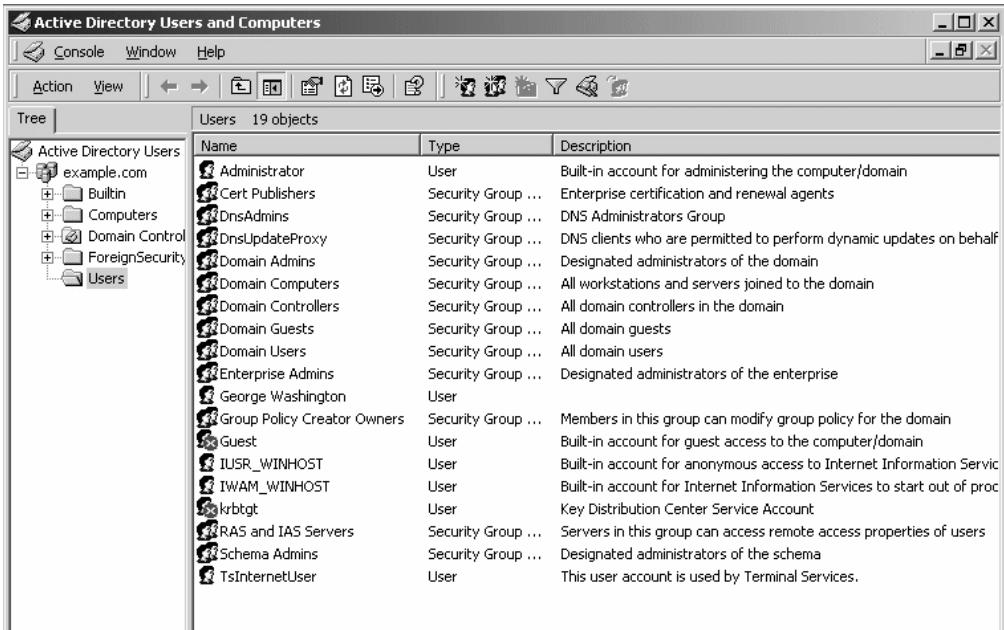
1.

From the Windows Start menu, select 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 right) and select Users.

You should see the *George Washington* entry as follows:

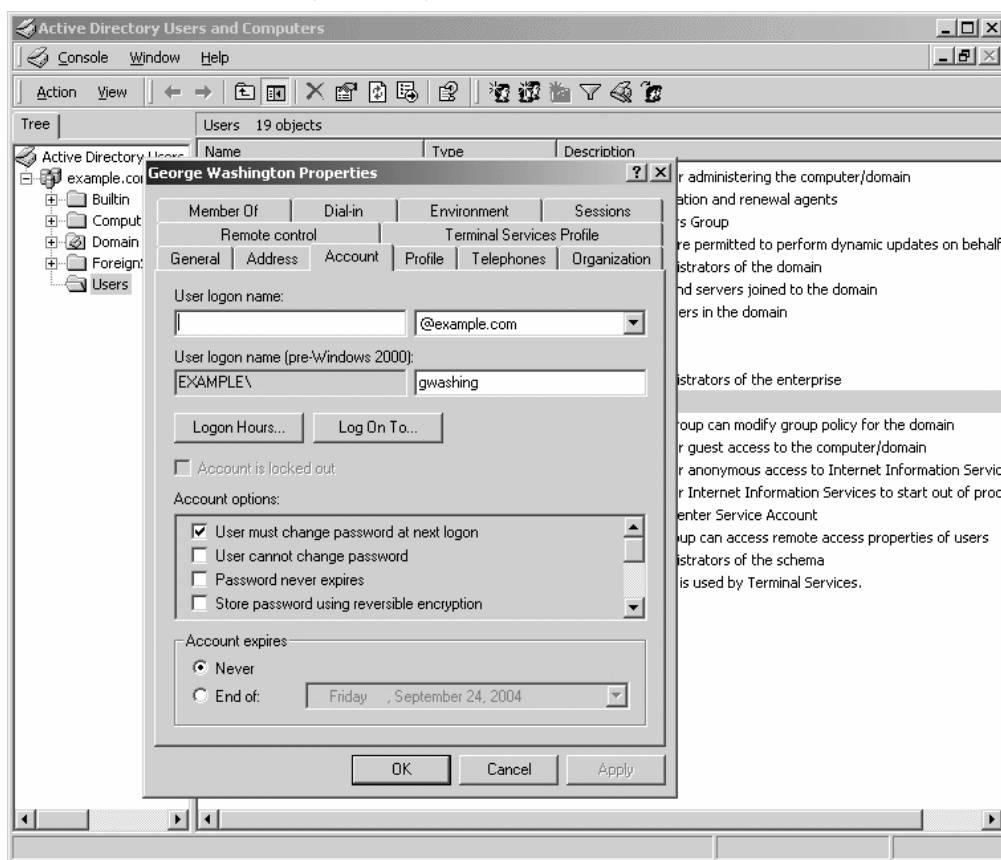
Figure A-12 Viewing the User on Windows



3. Right-click the George Washington entry and select Properties from the pop-up menu.

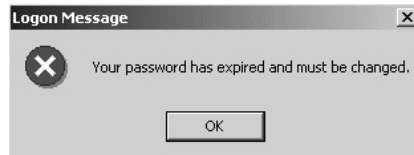
When the George Washington Properties dialog box is displayed (Figure A-13), check the Account options section and you can see that the User must change password at next logon check box is enabled, which means George will be required to change his password the next time he logs on.

**Figure A-13** George Washington's Properties



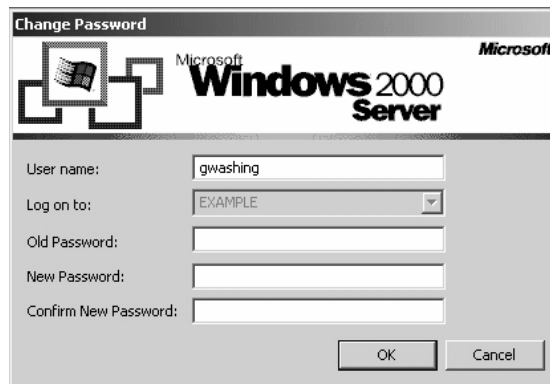
4. If you log in as George, you can see that Windows is correctly tracking the entry because the log-in attempt results in an expired password message prompt, as shown in Figure A-14.

**Figure A-14** Forcing a Password Change



5. Click OK to close the Logon Message dialog box, and the following Windows dialog box is displayed so you can provide a new password.

**Figure A-15** Change Password Dialog Box

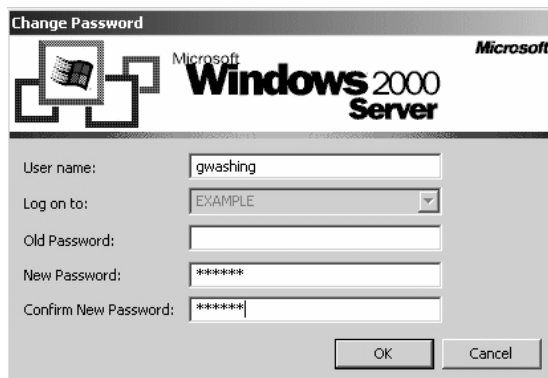


6. Enter and confirm a new password, but *do not* provide a value for the Old Password field (see Figure A-16).

Because this is first time the user has logged on (since being created over protocol), supplying an old password value will cause an error message and Windows will ask you to enter the new password again. After this first log-on, Windows will require the user to provide the old password.

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

**Figure A-16** Providing a New Password



If Windows accepts the new password, a message is displayed stating that the new password has been accepted (see Figure A-17).

**Figure A-17** Accepting the New Password



At this point, George's entry has moved from Case 3 (where the Windows entry is stale and the LDAP store is current) to Case 2 (where Windows is current and the LDAP store entry is stale).

George's entry will maintain this condition until the next time he binds to the LDAP store. At that time, the entry will move to the Case 1 (where the entry is current on both Windows and the LDAP store).

# Configuring Systems to Prevent Eavesdropping

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

Some of the required configuration changes are addressed when you configure Identity Synchronization for Windows. For example, on Windows (for Windows 2000 or later), the Windows' password policies require that all password changes must be made using secured methods. Consequently, simply configuring the system partially addresses the security requirement.

However, it is still possible for eavesdroppers to 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 to do so over TLS. For PAM clients, you must configure them to trust the LDAP store and ensure that `idsconfig` specifies `TLS:pam_ldap:simple` as the only authentication method for the LDAP store.

One final caveat — `root` accounts cannot use the `passwd` command arbitrarily to change an user's password on PAM client hosts. You might consider this restriction to be a limitation, it depends on whether you trust the PAM client administrators or not.

`/etc/pam.conf`

## Introducing Windows NT into the System

Previously in this appendix, it was stated that the term *Windows* referred to Windows platforms using Active Directory for authentication; however, the system being discussed in this appendix can use Windows NT in place of (or along with) newer variations of Windows.

Be aware however, 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. Consequently, when Identity Synchronization for Windows environments cohabitate with Windows NT they expect password changes to be captured at their source and at the time the change is made. This capture requirement has ramifications when you initially bring up an Identity Synchronization for Windows environment. Specifically, the Identity Synchronization for Windows system needs to see a password change for any entry before it can claim that the entry is synchronized.

Synchronization in an NT environment is similar to the processing steps described in “Case 3” on page 153, where you can modify 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 then neither of the two systems would contain stale passwords. However, because you created the passwords by deterministic means, it may be easy to guess these passwords.

With this technique, you can limit potential damage if you use Windows NT's password policy administration to force all users to change their password at their next logon. As each user changes their password, the Identity Synchronization for Windows password DLL installed on the Primary Domain Controller will forward 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 it in a production environment.

---

## Code Example A-1      Example /etc/pam.conf File

```
#
#ident  "@(#)pam.conf          1.2002/01/23  SMI"
#
# Copyright 1996-2004 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.
#
# 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
#
# 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
```

# Identity Manager and Identity Synchronization for Windows Cohabitation

This appendix describes how Sun Java System Identity Synchronization for Windows 1 2004Q3 and Identity Manager 5.0 SP 2 co-exist in a customer deployment as part of a larger user provisioning strategy, to facilitate native password changes on Directory Server and Active Directory.

This appendix augments Chapter 3, “Case Study: Deploying in a High-Availability Environment Over a Wide Area Network Using SSL”, and focuses on the integration of Identity Synchronization for Windows with Identity Manager, and the changes required to Identity Manager, for coexistence.

This appendix assumes knowledge of the concepts and deployment experience of Identity Manager. For details, see <http://docs.sun.com>.

This appendix includes:

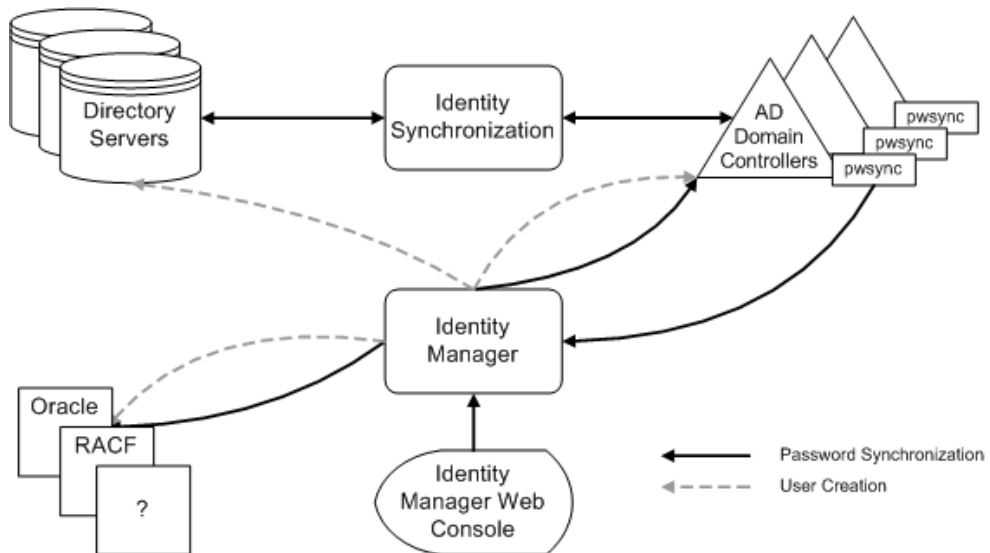
- “Overview” on page 164
- “Identity Manager and Identity Synchronization for Windows Functionality” on page 165
- “Password Changes on Active Directory” on page 166
- “Password Changes on Directory Server” on page 166
- “Password Changes and Provisions Originating from Identity Manager Console” on page 167
- “Configuring Identity Manager and Identity Synchronization for Windows” on page 167

# Overview

Figure B-1 illustrates the three important components of the Identity Manager and Identity Synchronization for Windows cohabitation deployment: Active Directory Domains, a separate Directory Server deployment, and any other Identity Manager-managed resource which does not include the previous two, for example, Oracle RDBMS.

The Identity Manager Console handles resource administration such as system-wide password changes and creation of users. 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 DLL installed on the Primary Domain Controllers of Windows systems). All password changes originating from the Identity Manager Administrator Console are subsequently propagated to all Identity Manager resources, except the Sun Java System Directory Server. All user creations originating from the Identity Manager Console are reflected to all resources, including Directory Servers. For details, see “Configuring `pwsync` to Not Propagate Passwords to Directory Server”.

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



# Identity Manager and Identity Synchronization for Windows Functionality

The effective functionality of Identity Manager and Identity Synchronization for Windows results from having both deployed and configured to function as a single system.

The Identity Synchronization for Windows functionality comprises:

- Detection of all 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:

- user creations
- user deletions
- non-password attributes

The Identity Manager functionality, in cohabitation with Identity Synchronization for Windows, comprises:

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

## Password Changes on Active Directory

Passwords modified on Active Directory are propagated through the Identity Manager-Identity Synchronization for Windows deployment as described below (See Figure B-1 for illustration):

1. The user resets the password on Active Directory by using the Change Password option in the Task Manager dialog 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 On Demand Synchronization in the *Sun Java™ System Identity Synchronization for Windows Installation and Configuration Guide*)
4. Identity Manager's `pwsync` command 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-Identity Synchronization for Windows deployment as described below:

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` command 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 Console

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

User creation originating from the Identity Manager Administration Console 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 “Handling Identity Manager-Provisioned Users”.

## Configuring Identity Manager and Identity Synchronization for Windows

Configuring Identity Manager and Identity Synchronization for Windows involves the following tasks:

1. “Setting Up Identity Manager 5.0 SP2 and Later”
2. “Setting Up Identity Manager 5.0 SP1 and Earlier”
3. “Configuring Identity Synchronization for Windows”

### Setting Up Identity Manager 5.0 SP2 and Later

Identity Manager 5.0 SP2 introduced a new form property that prevents the Directory Server resource from being shown as a resource where passwords can be changed. Identity Manager 5.0 SP 2 also introduced a new system configuration property that can be used to prevent `pwsync` from reflecting password changes to the Directory Server resource.

## Configuring the Form Property

To ensure that Identity Manager does **not** propagate user password changes to Directory Server but instead **only** propagates them to Active Directory, and then relies on Identity Synchronization for Windows to propagate them to Directory Server, the following form property can be added to any form used for changing a user's password. This will prevent a resource from being displayed in the table of resources where password changes occur.

```
<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 above, `name` (a string), or by `type` (also a string). The forms to which this property must be included are:

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



---

**NOTE** Some of the forms above already include the form property. In such scenarios, only the `new` attribute condition needs to be added from the XML fragment above.

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 condition above 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, it can be added by copying the full XML fragment above, or by adding 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` Plugin detects a password change. In an Identity Manager-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. This attribute can be added to the system configuration object by going to the `/debug` page (for example, <http://applicationserverhost:port/idm/debug>), listing objects of type `Configuration`, and editing the System Configuration to include the following

```
<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 there is more than one resource to exclude, use a comma-separated list.)

## Setting Up Identity Manager 5.0 SP1 and Earlier

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 workflow:

1. Install `pwsync` on all Active Directory domain controllers.
2. Modify the Identity Manager task definition for Change User Password, which can be done from 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 resource. This can be done 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, “Case Study: Deploying in a High-Availability Environment Over a Wide Area Network Using SSL”. Identity Synchronization for Windows should not be configured 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, and visa-versa. To establish this link for new users, administrators 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. Periodic automated execution is feasible using a scheduled UNIX cron job. For details, see "Periodic `idsync resync` Operation for Primary Installation" on page 115.

# Logging and Debugging

This appendix expands on the logging and troubleshooting information available in the *Sun Java™ System Identity Synchronization for Windows Installation and Configuration Guide* by describing how to use audit and debug logging to isolate problems in the system.

This appendix includes:

- Audit Logging and Action IDs
- Debug Logging
- Windows NT Change Detection
- Changing Central Logs File Location
- Changing Component Logs File Location
- Isolating Problems in Directory Server
- Isolating Problems in Message Queue
- Isolating Problems in Active Directory

## 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 will be logged to the central log.

---

<b>NOTE</b>	Changes to the log level are not propagated to subcomponents, that is, the Directory Server Plugin, 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.
-------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

---

## Actions

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 will 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 the 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 have been sent during an `idsync resync` operation. The source connector during a `idsync resync` operation sends one Sentinel action for each Synchronization User List that is processed, to signal that all users in that Synchronization User List have been 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`, while the other attributes are required, for example, `objectguid` or `pwdlastset`. Other attributes are sometimes referred to as meta attributes.

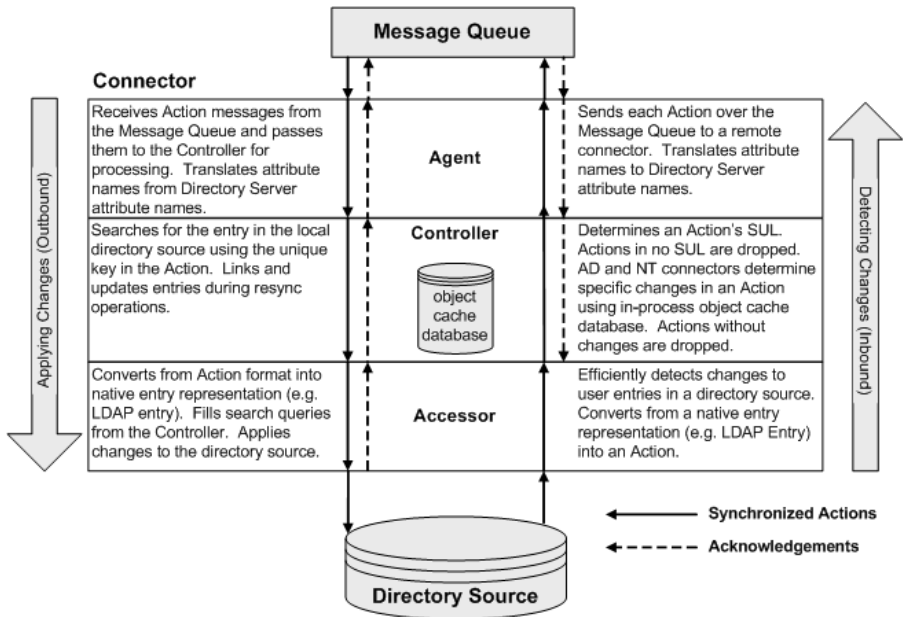
## Connector Layers - Accessor, Controller, and Agent

During processing, an action is passed between the 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:** 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 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 higher performance, the connector processes many actions at once, so log messages for different actions are 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 re-delivering 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 applies to log messages, a sample message from the central `audit.log`, when user John Test's password was changed, is given below:

```
[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 the 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 it through the system. A monotonically increasing sequence number (the SN value) also appears in the log entry. In rare circumstances where 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 dspswuserlink: 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 will include the Action ID, but some log entries will 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 above 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 usunchanged: 313464 whenchanged: 20041101151755.0Z
passwordchanged: TRUE dn: uid=jtest,ou=People, dc=gt,dc=com}." (Action
ID=CNN101-FFEF4A8986-206, SN=10)
[01/Nov/2004:07:17:56.567 -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 may be required to correlate log entries between the two files.

---

## Directory Server Plugin

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

- `nsslapd-infolog-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, then bit-wise OR in the 16th least significant bit into the value.

---

To enable additional logging, set the attributes on 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.

Using debug logging to isolate a problem can be a time-consuming process. Be sure to read the “Troubleshooting” chapter in the *Sun Java™ System Identity Synchronization for Windows Installation and Configuration Guide* before using this method.

## 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 below.

```
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 JVM option.

```
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 thirty seconds after stopping the Identity Synchronization for Windows daemon or service before restarting it. Once 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, requires additional processor cycles that can reduce throughput, and 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 log levels. The log levels for the debug logging behave identical to the setting 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.

Table C-1 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
accessor.level	Interaction with directory sources for detecting and applying changes. This is useful when problems accessing a directory source need to be diagnosed.	FINER
accessor.saint.level	Communication between the connector and the subcomponents.	FINE (if not specified, inherits from accessor.level)
controller.level	Processing that occurs within the Controller including determining membership in an SUL and interaction with the object cache.	FINER
agent.level	Processing that occurs within the agent including mapping attributes, sending messages over Message Queue, and receiving messages from Message Queue.	FINER
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)

**Table C-1**    Component-Level Debug Log Levels

Component	Type of Logging	Log.properties Level
centralLogger	Processing done by the Central Logger.	<default> (This can only be set in the CentralLoggerLog.properties)
crom.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.

<b>NOTE</b>	All messages that appear in the audit log file also appear in the debug log file to facilitate correlation of events between the logs.
-------------	----------------------------------------------------------------------------------------------------------------------------------------

## 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, the installer log files are written to the `/var/sadm/install/logs` directory.
- On Windows, 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
```



---

<b>NOTE</b>	Secure information such as passwords might appear in the installer log file when debug logging is enabled.
-------------	------------------------------------------------------------------------------------------------------------

---

## 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 which controls the amount of logging to generate. It 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

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

The Windows NT Connector periodically polls the Windows NT subcomponents (Identity Synchronization for Windows NT Change Detector Service 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, then the subcomponents can accumulate a significant number of changes in memory. (A typical change record requires less than 100 bytes of memory, so 10 MB of memory can accommodate 100,000 changes. However, this scenario is not likely to occur on Windows NT).

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(TM) System Identity Synchronization for Windows\1.1, you can view the following registry values:

```
"PfLastAckedChangeNumber"=hex:05,00,00,00,00,00,00,00
"PfLastAckedChangeTime"=hex:54,64,86,41
"PfHighestChangeNumber"=hex:05,00,00,00,00,00,00,00
"PfHighestChangeTime"=hex:52,64,86,41
```

Where:

- **PfLastAckedChangeNumber:** Specifies the last change number the connector acknowledged to the Password Filter.
- **PfLastAckedChangeTime:** 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 PfLastAckedChangeNumber, then 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=CNN102-FFF4EFF25B-1162, SN=0)
```

In this example, the `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 the `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` in the PsTools software bundle from <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:
...

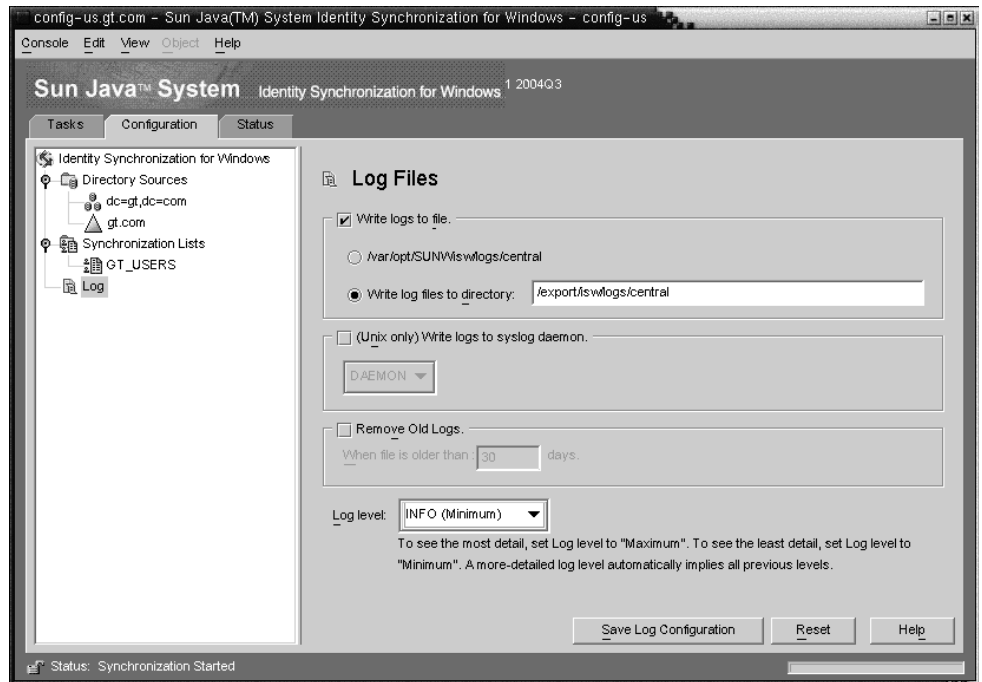
```

In this example, the log record number is 27 (which corresponds to the value in `LastProcessedSecLogRecordNumber - 0x1b`).

# Changing Central Logs File Location

The location of the central logs can be changed using the console on the Log Configuration tab.

**Figure C-2** Log Configuration Tab



# Changing Component Logs File Location

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/SUNWiw/logs/CNN101` to `/export/isw/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/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 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/SUNWiw/lib:/usr/lib/mps/secv1
-Djava.util.logging.config.file=/var/opt/SUNWiw/resources/Log.properties
-Dcom.sun.directory.wps.logging.directory=/export/isw/logs/adConnector
-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 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 changes to take effect.

## Isolating Problems in Directory Server

For details on concepts 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 on understanding and configuring the Message Queue Broker's 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:

- 1 = minimum
- 3 = medium
- 5 = maximum

To isolate problems with Identity Synchronization for Windows, the values to set are Directory Access and LDAP Interface Events.



# Glossary

The following terms are used in the Identity Synchronization for Windows product and throughout this documentation set.

**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 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** Holds descriptive information about an entry. Attributes have 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** 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.

Each connector produces an audit log of the users processed by that connector, and there is a centralized audit log containing an aggregation of the audit logs produced by all of the connectors in your deployment.

**authentication** Process of proving the identity of the client user to Directory Server. Users 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 objectclass** An objectclass 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.

**base distinguished name** See base DN.

**bind DN** Distinguished name used to authenticate to an LDAP directory (e.g. Active Directory or Directory Server) when performing an operation.

**bind distinguished name** See bind DN.

**Broker** See Sun Java System Message Queue Broker.

**CA** See Certificate Authority.

**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. It 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. You can view the central logs directly or from the Identity Synchronization for Windows Console. By default, the central logs are available on the machine where Core was installed under 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 you configure Identity Synchronization for Windows Connectors to use SSL communication, you must add certificates to the connector's certificate databases before trusted SSL communication can occur. See also Certificate Authority.

**Certificate Authority** A company or organization that sells and issues authentication certificates. You may purchase an authentication certificate from a Certificate Authority (also known as a CA) that you trust. A root Certificate Authority certificate is used to sign other certificates. When configuring an Identity Synchronization for Windows Connector to use SSL, you 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: `cert8.db`, `key3.db`, and `secmod.db`. In Identity Synchronization for Windows, each connector has its own certificate database directory (for example, `<install-root>/etc/CNN100`). See also certificate.

**character type** Distinguishes alphabetic characters from numeric (or other) characters and the mapping of upper-case to lower-case letters.

**CLI** See command line interface

**client** See LDAP client.

**command line interface** A means of communication between a program and its user, based solely on textual input and output. Commands are input with the help of a keyboard or similar device, and are interpreted and executed by the program. The Identity Synchronization for Windows command line interface is named `idsync` and is available in the `bin/` directory where you installed Core.

**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 instance 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 Identity Synchronization for Windows' interaction with a single data source (such as a Directory Server, an Active Directory domain, or a Windows NT domain). A connector is responsible for detecting user changes in the data source and publishing these changes to remote connectors over Message Queue, and for subscribing to user change topics and applying updates from these topics to the data source.

**console** A Graphical User Interface used to configure and monitor server applications. The 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. You 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 machine 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** 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** The privileged directory server administrator, comparable to the root user in UNIX. 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.

**DM** See Directory Manager.

**domain** (1) (n.) The last part of a fully qualified domain name that identifies the company or organization that owns the domain name (for example, example.com, host.example.com).

(2) (n.) Resources under control of a single computer system.

**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.

**DNS** Domain Name System. System used by machines on a network to associate standard IP addresses (such as 198.93.93.10) with hostnames (such as www.example.com). Machines normally get the IP address for a hostname from a DNS server or look up the address in tables maintained on their systems.

**file extension** Portion of a filename following the period or dot (.) that typically defines the file type (for example, .GIF and .HTML). For example, in a file named `index.html` the file extension is *html*.

**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 ASCII text format. File types are usually identified by the file extension (for example, .GIF or .HTML).

**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 control with the PDC emulator role, Identity Synchronization for Windows use this domain controller for synchronization. Otherwise, synchronization with the Sun Java System Directory Server might be delayed for several minutes.

**global catalog** A Windows repository that stores Active Directory directory topology and schema information for Active Directory directories.

**hostname** A name for a machine in the form `machine.domain.com`, which is translated into an IP address. For example, `www.example.com` is the machine *www* in the subdomain *example*, and domain *com*.

**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, 192.168.2.1).

**ISO** International Standards Organization.

**Java Message Service** A messaging standard API that allows application components based on the Java 2 Platform, Enterprise Edition (J2EE) to create, send, receive, and read messages. It enables distributed communication that is loosely coupled, reliable, and asynchronous.

**JMS** See Java Message Service.

**LDAP** Lightweight Directory Access Protocol. 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.

**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** Provides the means of locating directory servers using DNS and then completing the query via LDAP. A sample LDAP URL is `ldap://ldap.example.com`

**Lightweight Directory Access Protocol** See LDAP.

**locale** Identifies the collation order, character type, monetary format, and time / date format used to present data for users of a specific region, culture, and/or custom. This includes information on 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.

**MQ** See Sun Java System Message Queue.

**multimaster replication** 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 directory server masters. 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** (also known as root suffix) A specific suffix of a directory information tree (DIT) that is identified by its distinguished name (DN), e.g. `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.

**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 entry contains. For example, Directory Server specifies an `inetorgperson` object class which 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. This simplifies password synchronization in Active Directory environments.

**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 machines that stores UNIX user login names, passwords, and user ID numbers. It is also known as `/etc/passwd`, 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** An accessory program that can be loaded and then used as part of the overall system.



For example, Identity Synchronization for Windows uses the Directory Server Plugin to enhance Directory Server Connector change-detection features and to provide bidirectional support for password synchronization between Active Directory and Directory Server.

**preferred directory server** A directory server master 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 server masters.

**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** A Directory Server database (cn=changelog) 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 machines (also called superuser). The root user has complete access privileges to all files on the machine. 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 the directory. When information that does not match the schema is stored in the directory, clients attempting to access the directory may 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 master 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 server or any other directory server master will not be synchronized until the preferred directory server is available.

**Secure Sockets Layer** See SSL.

**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. 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 LFAP.

**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 `user`, and the default Directory Server object class is `inetorgperson`. See Auxiliary objectclass.

**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 in a remote machine or separate process. The subcomponent communicates with 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 Plugin, 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.

**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. The 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 standalone Java server that provides clients access to the Sun Java System Message Queue. On Solaris, the Broker is controlled via the `/etc/init.d/imq` daemon script, and on Windows, it is controlled via the "iMQ Broker" service. Identity Synchronization for Windows configures and starts the broker during Core installation.

**superuser** See root.

**synchronization host** Servers that store synchronized data according to the rules defined in the Synchronization User Lists (SULs).

**Synchronization User List** Defines users in the Sun and Windows directories to be synchronized. A Synchronization User List 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, the Watchdog is controlled via the /etc/init.d/isw daemon script, and on Windows, it is controlled via the "Sun Java™ System Identity Synchronization for Windows" service.

## SYMBOLS

%TEMP% environment variable 184

## A

accessor connector layers 175

accessor.level 183

accessor.saint.level 183

accounts, new user 27

action ID types 174

actions 174, 175, 177, 179, 193, 198, 200

Active Directory

    changing passwords 166

    configuring 35

    connector layers 175

    deleted entries 27

    deployment considerations 19–21

    domains 25

    failover configuration 93

    Global Catalog 35

    global setting 94

    inactive accounts 27

    new users 27

    password changes 166

    samaccountname 25

    special users 29

    synchronization user lists 46

activedirectorydomainname attribute 68

administration rights 29

agent connector layers 175

agent.level 183

agentout.level 183

attributes

    activedirectorydomainname 20, 68

    configuring 42

    data 175

    destinationIndicator 58, 64, 65, 67, 68

    dspswuserlink 27, 65, 71

    dspswvalidate 65

    mapping 43

    mappings 43

    modification settings 41

    nsslapd-infolog-area 180

    nsslapd-infolog-level 180

    other 175

    pwdlastset 177

    samaccountname 25, 60, 82

    shadowmax 46

    shadowmin 46

    shadowwarning 46

    uid=PSWConnector 29

    user\_name 25

    user\_nt\_domain\_name 68

    userPassword 66, 67

audit logging 173–192

audit.log 176

authentication

    Active Directory 133

    denying 147

    description 194

    domain controllers 127

    editing /etc/nsswitch.conf 147

- editing /etc/pam.conf 146, 147, 148
- for failover 127
- on Solaris 24, 134, 144
- on Windows 139
- over TLS 158
- rules 146
- using PAM 134, 144, 147
- authentication certificates 194, 195
- auxiliary object classes 194

## B

- Base DNs 138
- brokers, Message Queue 111, 112, 113, 115, 203

## C

- Central Logger
  - description 195
  - enabling debug logging 181
  - log locations 190
  - out-of-order messages 177
  - plugin logging 129
  - processing logs 184
- central logs 179, 189, 194
- centralLogger component 184
- Certificate Authority 21, 101, 195
- certificate databases 21, 101, 106, 195
- certificates
  - Active Directory 101
  - Active Directory SSL 21
  - adding to certificate databases 106
  - authentication 194
  - Certificate Authority 21, 101
  - component requirements 104
  - configuring connectors 158
  - definition 195
  - Directory Server 101, 106
  - trusted 84, 91, 101, 104
  - using Certificate Authorities 101
  - using with plugins 91

- using with SSL 21, 84, 91, 101
- change detection
  - action paths 175, 198, 200
  - Active Directory 70, 115, 121, 169
  - connectors 21, 33, 86, 121, 122, 123, 126, 196
  - controller 175, 196
  - Directory Server 21, 33, 70, 86, 123, 201
  - failover 126
  - failures 32, 126
  - Identity Manager 165, 167
  - Identity Synchronization for Windows 122, 165, 166, 174
  - pwsync command 166
  - SUL filters 47, 54
  - Windows NT 185
  - Windows NT Change Detector 174, 185
- Change Detector, Windows NT 174
- commands
  - idsync certinfo 101, 104–105
  - idsync printstat 96, 99, 124, 125, 181
  - idsync resync 20, 57–68, 77, 83, 89, 106, 111–121, 126, 172, 179
  - idsync startsync 96, 100, 129
  - idsync stopsync 128
- common.level 184
- components
  - changing logs location 190
  - Core 195, 203, 204
  - logging 183
  - trusting certificates 104
- configuring
  - Active Directory 35
  - attribute mappings 43
  - Core 33
  - Directory Server 33, 80
  - failover domain 38
  - Identity Manager 122
  - Identity Synchronization for Windows 68
  - preferred Directory Server 97
  - shadowAccount object class 45
  - synchronization settings 41
  - synchronization user lists 46
  - Windows NT 40
- connectors
  - changing log locations 190
  - definition 196

- initializing connector state 123, 124
- installing 57, 85, 110
- layers 175
- modification rate 20
- persisting states 125
- security requirements 21, 84, 158
- uninstalling 110
- updating 121
- using in large deployments 89
- WAN deployments 69
- console.log 185
- consoles
  - debug logging 185
  - Server Console 202
  - validations 54
- controller connector layers 175
- controller.level 183
- conventions used in book 14
- Core
  - components 195, 203, 204
  - configuring 33
  - configuring Directory Server 90
  - description 196
- CREATE actions 174
- creating users 27, 29
- credentials
  - Active Directory 127
  - bind 152, 158
  - candidate 159
  - Directory Manager 197
  - problems 110
  - proxy 145
  - specifying 36, 145
- crom.level 184

## D

- daemon, Identity Synchronization for Windows 108, 110, 128, 181, 182, 197
- data attributes 175
- databases
  - certificate 21, 101, 106, 195
  - object cache 115, 121, 124, 126, 177, 200

- retro changelog 201
- suffixes 203
- debug logging 111, 173–192
- debug page 169, 170
- debug.log 182
- debugErrors.log 182
- decryption routines 183
- DELETE actions 174
- deleting users 27
- deployment
  - examples 24, 80
  - HA environments 79
  - increasing connector worker threads 106
  - large deployments 89
  - performance requirements 20
  - preparing for 19
  - primary/failover configurations 108
  - running idsync resync 57, 111
  - synchronization direction 20
  - WAN 69
  - with Identity Manager 163–172
- destinationIndicator attribute 58, 64, 65, 67, 68
- directory resources, deployment example 24
- Directory Server
  - changing passwords 166
  - configuring 33, 80
  - configuring LDAP repository for PAM 136–139
  - connector 57
  - connector layers 175
  - deleted entries 27
  - deployment considerations 19–21
  - deployment example 24
  - hierarchical DIT 28
  - idsync resync 54
  - in HA environment over WAN 79–131
  - in MMR environments 24–77
  - inactive accounts 27
  - isolating problems 191
  - LINK actions 174
  - masters 21
  - new users 27
  - PAM 24
  - plugin installation 57
  - populating LDAP repository for PAM 140–143
  - preferred source 34
  - preventing password change propagation 169

- product documentation links 17
- purpose 24
- secondary server 34
- uid 25
- using with Identity Manager 167–171
- verifying synchronization 152–157
- Directory Server Plugin 21, 129, 174, 180
- directory sources, specifying 34
- disabled accounts 27
- distinguished names 194
- DNS 137, 138, 145, 146, 197
- documentation
  - audience 13
  - overview 16
  - recommended reading 16
- domain controllers 170
  - architecture example 24
  - authenticating 127
  - configuring 98
  - definition 197
  - deployment example 80–130
  - failover 38
  - for WAN deployment 69
  - logs 192
  - PDC 159, 164
  - PDC FSMO role owners 37, 130
  - primary 40
  - uSNChanged values 113
  - with SSL 101, 104
- dspswuser auxiliary object class 65
- dspswuserlink attribute 27, 65, 71
- dspswvalidate attribute 65

## E

- encryption keys 108, 129, 131
- encryption.level 183
- environment variables
  - %TEMP% 184
  - ISW\_DEBUG\_INSTALL 184
- environments
  - Active Directory 133
  - changing passwords 26, 83

- high-availability over WAN 79
- Identity Synchronization for Windows and Identity Manager cohabitation 163
- MMR 23–77, 201, 202
- NT 133
- Solaris 8 136
- using PAM 134
- WAN 69
- Windows NT 159
- Example Bank deployment example 23–77
- executing
  - periodic resynchronization 115

## F

- failover
  - Active Directory connector 124
  - Directory Server connector 123
  - domain controllers 38
  - installation 87, 97, 108, 121, 126, 129, 131
  - maintenance 126
  - monitoring logs 131
  - starting synchronization 129
  - stopping synchronization 128
- failures, detecting 122
- features 19, 28
  - DIT 28
  - moving users 28
  - SUL filters 28
  - WAN deployments 28
- filters, using 47, 54, 89, 112
- form property, configuring 168

## G

- Global Catalog 35, 36, 198
- global setting, Active Directory 94
- Global Telco
  - configuration 84
  - configuring Active Directory 92
  - configuring Directory Server 90



- on-demand synchronization 92
- primary installation 84
- requirements 83
- Globally Unique Identifier. *See GUIDs*
- glossary of terms 193
- GUIDs 26, 27, 174

## H

- High Encryption Pack 19
- high-availability environments 79–131
- hub replicas 21, 81

## I

- Identity Manager
  - cohabitation with Identity Synchronization for Windows 163–172
  - Global Telco case study 80, 115
  - password changes 83, 167
- Identity Synchronization for Windows
  - failover installation 87, 97, 108, 121, 129, 131
  - installing and configuring 68
  - primary installation 85, 90, 110, 112, 115, 122, 128, 131
- idsconfig 137, 138, 145, 158
- idsync certinfo command 101, 104–105
- idsync command line interface 196
- idsync printstat command 96, 99, 124, 125, 181
- idsync resync command
  - central log file 179
  - deployment example 57–68
  - failover maintenance 126
  - increasing connector worker threads 106
  - initial operation 111–115
  - linking users 20, 54, 68, 77, 83, 89, 172
  - primary installation 115–121
  - reducing peak load 89
  - REFRESH action 174
- idsync startsync command 96, 100, 129
- idsync stopsync command 128

- inactive accounts 27
- increasing connector work threads 106
- initializing connector state 123
- installation
  - connectors 57, 85
  - failover 97
- ISW\_DEBUG\_INSTALL environment variable 184

## L

- LANs 69
- layers, connector 175
- LDAP
  - accessor 175
  - attributes 46
  - communicating with domain controllers 86
  - deployment considerations 19
  - filters 89, 112
  - maximum connections 108
  - PAM 24, 26, 45, 68, 133–162
  - sample URL 199
  - shadowAccount object class 45
- LINK actions 174
- linking users
  - idsync resync 57
  - new users 89
- log.properties level 183
- logging
  - audit and debug 173–192
  - central logs 189
  - centralLogger 184
  - component logs 190
  - components 183
  - configuring Directory Server plugin 180
  - console logs 185
  - debug 111
  - levels 111
  - message samples 176
  - monitoring logs 131
  - security logs 188
  - types 183
- login IDs 26

## M

- maintenance, failover 126
- manager.level 183
- Message Queue
  - agent 175
  - broker description 203
  - broker memory limits 111, 112, 113, 115
  - isolating problems 192
- Microsoft
  - Knowledge Base Articles 17
  - publications 17
- migrating
  - Example Bank case study 25
  - from NT to Active Directory 67, 70
  - user passwords 75
- MMR environments 23–77, 84, 201, 202
- modification rate 20
- MODIFY actions 174
- monitoring logs 131
- mq.level 183
- Multi-Master Replication. *See MMR environments*
- multiple object classes 28
- multiple passwords, setting 109

## N

- new users
  - Active Directory 27
  - creating 20
  - Directory Server 27

## O

- object cache databases 115, 121, 124, 126, 177, 200
- object classes
  - auxiliary 194
  - dspswuser 65
  - shadowAccount 45, 68
- on-demand synchronization 92

## P

- PAM
  - configuring LDAP repository 136–139
  - Directory Server 24
  - populating LDAP repository 140–143
  - verifying synchronization 152–157
- Password Filter DLL 174, 185
- Password Policies, Windows 17
- passwords
  - changing on Active Directory 166
  - changing on Directory Server 166
  - changing through Identity Manager 167
  - dspswvalidate attribute 65
  - encrypting 86, 108
  - Example Bank's requirements 26
  - losing changes 108
  - managing on Solaris 24
  - on-demand synchronization 38, 86, 99, 105, 121, 126
  - preventing display in command line 116
  - preventing propagation 169
  - propagating changes 83
  - replicating to PDC FSMO role owner 37
  - resetting 116
  - running idsync resync 111
  - setting multiple 109
  - shadowmax attribute 46
  - shadowmin attribute 46
  - shadowwarning attribute 46
  - specifying 36
  - synchronizing 19, 26, 58, 83
  - userPassword attribute 65, 67
- PDC 25, 159, 164
- PDC FSMO role owners 37, 69, 82, 92, 98, 130
- peak modification rate 20
- periodic resynchronization 115, 121
- permissions 31, 116, 194, 200
- Pluggable Authentication Module. *See PAM*
- plugins
  - Directory Server 21, 57, 129, 174, 180
  - Retro Changelog 21
- preparing for deployment 19
- Primary Domain Controller. *See PDC*
- primary installation
  - configuring Directory Server 90

- Identity Synchronization for Windows 85, 90, 110, 112, 115, 122, 128, 131
- provisioned users 172
- psloglist.exe 188
- publications
  - Microsoft 17
  - related 16
- pwdlastset attribute 177
- pwsync, configuring 169

## R

- read-only replicas 21, 81
- recommended reading 16
- REFRESH actions 174
- reinstalling Directory Server plugins 129
- related documentation 16
- replicas
  - hub 21, 81
  - read-only 21, 81
- replication agreements 81
- requirements
  - Example Bank 26
  - Global Telco 83
- resync interval 201
- resync.log 179
- resync-batch.pl script 113
- resyncDebug.log 182
- resynchronization 111
- retro changelog 123, 201
- Retro Changelog Plugin 21
- rights, administration 29
- role owner, PDC FSMO 130

## S

- samaccountname attribute 60, 82
- secondary servers 34
- Secure Sockets Layer. *See* SSL

- security
  - configuring 91, 158
  - considerations 21, 183
  - logs 188
  - using TLS 135
- SENTINEL actions 174
- Server Console 202
- setting up
  - Identity Manager 170
  - SSL 101
- Shadow Account
  - shadowmax 46
  - shadowmin 46
  - shadowwarning 46
- shadowAccount auxiliary object class 45, 68
- shell prompt conventions 16
- special users
  - administration rights 29
  - creating 29
  - delegating control 29
  - permissions 31
- specifying
  - credentials 36
  - directory sources 34
  - passwords 36
  - secondary servers 34
- SSL
  - certificates 21
  - enabling 19, 21
  - requirements 21, 33, 84
  - setting up 101–106
  - with trusted certificates 21, 84, 91, 101
- starting synchronization failover 129
- suffixes 203
- SULs
  - description 203
  - determining membership 175
  - Directory Server parameters 52
  - moving users 27
  - purpose 26
  - using filters 47, 54
  - validating filters 54
- Sun Java™ System Directory Server. *See* Directory Server

Sun Java™ System Identity Synchronization for Windows. *See Identity Synchronization for Windows*

Sun Java™ System Message Queue. *See Message Queue*

symbol conventions 15

synchronization

starting 129

stopping 128

synchronization settings 41

attribute mapping 43

attribute settings 42

configuring 41

shadowAccount object class 45

Synchronization User Lists. *See SUL*

synchronizing

inactive accounts 27

passwords 19, 26

users 27

SYNCING state 124

System Manager

checking status 96, 100, 125

component logs location 190

description 203

enabling debug logging 181

new configurations 108

processing logs 183

## T

tasks.level 184

TLS 135, 158

Transport Layer Security. *See TLS*

troubleshooting 173

trust requirements 104

trusted certificates 21, 84, 91, 101, 104

typographic conventions 14

## U

UNKNOWN actions 174

unlinking migrated users 71

user accounts, eliminating 26

user passwords

Change My Password Form 168

Change Password Form 168

Change User Password Form 168

changing on Active Directory 166

changing on Directory Server 166

changing through Identity Manager 167

dspswvalidate attribute 65

encrypting 86, 108

Example Bank 26

losing changes 108

migrating 75

modifying 125

on-demand synchronization 38, 86, 99, 105, 121, 126

preventing display in command line 116

preventing propagation 169

propagating changes 83, 166, 168

replicating to PDC FSMO role owner 37

Reset User Password Form 168

resetting 116

running idsync resync 111

setting multiple 109

shadowmax attribute 46

shadowmin attribute 46

shadowwarning attribute 46

synchronization direction 19

synchronizing changes 42, 58, 83

userPassword attribute 65, 67

user\_nt\_domain\_name attribute 68

userPassword attribute 66, 67

users

creating 27, 29

deleting 27

linking migrated 75

moving between domains 76

provisioned 172

synchronizing 27

unlinking migrated 71

uSNChanged values 113

## **W**

WAN deployments 69, 79

websites

- Directory Server publications 17

- Identity Synchronization for Windows  
publications 16

- Message Queue publications 17

Windows

- Global Catalog 36, 198

- Password Policies 17

Windows NT

- Change Detector 174

- configuring 40

- connector layers 175

- Password Filter DLL 174

- PDC 40

- SUL\_NT 46

- synchronization user lists 46

- user\_name 25

## **X**

xml

- form properties 168, 169

- link files 60

- linkusers-ad-only.cfg 62, 63

xml.level 184

