Tivoli® Directory Integrator
Version 7.1

Password Synchronization Plug-ins Guide

SC27-2708-00

Tivoli® Directory Integrator
Version 7.1

Password Synchronization Plug-ins Guide

SC27-2708-00

Note

Note: Before using this information and the product it supports, read the general information under Appendix C, "Notices," on page 93.

Product Version 7.1

This edition applies to version 7.1 of the IBM Tivoli Directory Integrator and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Preface

This document describes the procedural steps that are required to achieve password synchronization between IBM® Tivoli® Directory Integrator and a number of IBM and third party products.

The chapters in this guide cover all the plug-ins available for password synchronization at the time of publication. Please see the IBM Tivoli Directory Integrator Web site for any later plug-ins or updates.

This document assumes that both Tivoli Directory Integrator and the products to be integrated are already installed, configured and running on your network. No details are provided regarding the installation and configuration of these products, except where necessary to achieve integration.

Who should read this book

Tivoli Directory Integrator components are designed for network administrators who are responsible for maintaining user directories and other resources.

This document assumes that you have practical experience installing and using both IBM Tivoli Directory Integrator and the product to be integrated.

Publications

Read the descriptions of the IBM Tivoli Directory Integrator library and the related publications to determine which publications you might find helpful. After you determine the publications you need, refer to the instructions for accessing publications online.

IBM Tivoli Directory Integrator library

The publications in the Tivoli Directory Integrator library are:

IBM Tivoli Directory Integrator V7.1 Getting Started
A brief tutorial and introduction to Tivoli Directory Integrator 7.1. Includes examples to create interaction and hands-on learning of IBM Tivoli Directory Integrator.

IBM Tivoli Directory Integrator V7.1 Installation and Administrator Guide
Includes complete information about installing, migrating from a previous version, configuring the logging functionality, and the security model underlying the Remote Server API of IBM Tivoli Directory Integrator. Contains information on how to deploy and manage solutions.

IBM Tivoli Directory Integrator V7.1 Users Guide
Contains information about using IBM Tivoli Directory Integrator 7.1. Contains instructions for designing solutions using the Tivoli Directory Integrator designer tool (**ibmditk**) or running the ready-made solutions from the command line (**ibmdisrv**). Also provides information about interfaces, concepts and AssemblyLine creation.

IBM Tivoli Directory Integrator V7.1 Reference Guide
Contains detailed information about the individual components of IBM

Tivoli Directory Integrator 7.1: Connectors, Function Components, Parsers and so forth – the building blocks of the AssemblyLine.

IBM Tivoli Directory Integrator V7.1 Problem Determination Guide
Provides information about IBM Tivoli Directory Integrator 7.1 tools, resources, and techniques that can aid in the identification and resolution of problems.

IBM Tivoli Directory Integrator V7.1 Messages Guide
Provides a list of all informational, warning and error messages associated with the IBM Tivoli Directory Integrator 7.1.

IBM Tivoli Directory Integrator V7.1 Password Synchronization Plug-ins Guide
Includes complete information for installing and configuring each of the five IBM Password Synchronization Plug-ins: Windows Password Synchronizer, Sun Directory Server Password Synchronizer, IBM Directory Server Password Synchronizer, Domino Password Synchronizer and Password Synchronizer for UNIX® and Linux®. Also provides configuration instructions for the LDAP Password Store and JMS Password Store.

IBM Tivoli Directory Integrator V7.1 Release Notes
Describes new features and late-breaking information about IBM Tivoli Directory Integrator 7.1 that did not get included in the documentation.

Related publications

Information related to the IBM Tivoli Directory Integrator is available in the following publications:

- IBM Tivoli Directory Integrator 7.1 uses the JNDI client from Sun Microsystems. For information about the JNDI client, refer to the *Java™ Naming and Directory Interface™ Specification* on the Sun Microsystems Web site at http://java.sun.com/j2se/1.5.0/docs/guide/jndi/index.html.
- The Tivoli Software Library provides a variety of Tivoli publications such as white papers, datasheets, demonstrations, redbooks, and announcement letters. The Tivoli Software Library is available on the Web at: http://www.ibm.com/software/tivoli/library/
- The *Tivoli Software Glossary* includes definitions for many of the technical terms related to Tivoli software. The *Tivoli Software Glossary* is available on the World-Wide Web, in English only, athttp://publib.boulder.ibm.com/tividd/glossary/tivoliglossarymst.htm

Accessing publications online

The publications for this product are available online in Portable Document Format (PDF) or Hypertext Markup Language (HTML) format, or both in the Tivoli software library: http://www.ibm.com/software/tivoli/library.

To locate product publications in the library, click the **Product manuals** link on the left side of the Library page. Then, locate and click the name of the product on the Tivoli software information center page.

Information is organized by product and includes READMEs, installation guides, user's guides, administrator's guides, and developer's references as necessary.

Note: To ensure proper printing of PDF publications, select **Fit to page** in the Adobe Acrobat Print window (which is available when you click **File->Print**).

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use information technology products successfully. With Tivoli Directory Integrator 7.1, you can use assistive technologies to hear and navigate the interface. After installation you also can use the keyboard instead of the mouse to operate all features of the graphical user interface.

Accessibility features

The following list includes the major accessibility features in Tivoli Directory Integrator 7.1:

- Supports keyboard-only operation.
- Supports interfaces commonly used by screen readers.
- Discerns keys as tactually separate, and does not activate keys just by touching them.
- Avoids the use of color as the only way to communicate status and information.
- Provides accessible documentation.

Keyboard navigation

This product uses standard Microsoft® Windows® navigation keys for common Windows actions such as access to the File menu, copy, paste, and delete. Actions that are unique to Tivoli Directory Integrator use Tivoli Directory Integrator keyboard shortcuts. Keyboard shortcuts have been provided wherever needed for all actions.

Interface Information

The accessibility features of the user interface and documentation include:

- Steps for changing fonts, colors, and contrast settings in the Configuration Editor:
  1. Type `Alt-W` to access the Configuration Editor **Window** menu. Using the downward arrow, select **Preferences...** and press `Enter`.
  2. Under the **Appearance** tab, select **Colors and Fonts** settings to change the fonts for any of the functional areas in the Configuration Editor.
  3. Under **View and Editor Folders**, select the colors for the Configuration Editor, and by selecting colors, you can also change the contrast.
- Steps for customizing keyboard shortcuts, specific to IBM Tivoli Directory Integrator:
  1. Type `Alt-W` to access the Configuration Editor **Window** menu. Using the downward arrow, select **Preferences...** .
  2. Using the downward arrow, select the General category; right arrow to open this, and type downward arrow until you reach the entry **Keys**.

     Underneath the **Scheme** selector, there is a field, the contents of which say "type filter text." Type `tivoli directory integrator` in the filter text field. All specific Tivoli Directory Integrator shortcuts are now shown.
  3. Assign a keybinding to any Tivoli Directory Integrator command of your choosing.
  4. Click **Apply** to make the change permanent.

The Configuration Editor is a specialized instance of an Eclipse workbench. More detailed information about accessibility features of applications built using Eclipse can be found at http://help.eclipse.org/help33/topic/org.eclipse.platform.doc.user/concepts/accessibility/accessmain.htm

- The information center and its related publications are accessibility-enabled for the JAWS screen reader and the IBM Home Page Reader. You can operate all documentation features using the keyboard instead of the mouse.

Vendor software

The IBM Tivoli Directory Integrator installer uses the InstallAnywhere 2009 (IA) installer technology.

The IBM Tivoli Directory Integrator 7.1 installer has accessibility features that are independent from the product. The installer supports 3 UI modes:

GUI
Keyboard-only operation is supported in GUI mode, and the use of a screen reader is possible. In order to get the most from a screen reader, you should use the Java Access Bridge and launch the installer with a Java access Bridge enabled JVM, for example:

```
install_tdiv71_win_x86.exe LAX_VM "Java_DIR/jre/bin/java.exe"
```

The JVM used should be a Java 6 JRE.

Console
In console mode, keyboard-only operation is supported and all displays and user options are displayed as text that can be easily read by screen readers. Console mode is the suggested install method for accessibility.

Silent
In silent mode, user responses are given through a response file, and no user interaction is required.

Related accessibility information

Visit the *IBM Accessibility Center* at http://www.ibm.com/able for more information about IBM's commitment to accessibility.

Contacting IBM Software Support

Contact IBM Software Support by using the methods described in the *IBM Software Support Guide* at the following Web site:

http://techsupport.services.ibm.com/guides/handbook.html

The guide provides the following information:

- Registration and eligibility requirements for receiving support
- Telephone numbers, depending on the country in which you are located
- A list of information you should gather before contacting customer support

A list of most requested documents as well as those identified as valuable in helping answer your questions related to IBM Tivoli Directory Integrator can be found at http://www-01.ibm.com/support/docview.wss?rs=697 &uid=swg27009673.

For more information, see Appendix B, “IBM Software Support,” on page 89.

Tivoli technical training

For Tivoli technical training information, refer to the IBM Tivoli Education Web site: http://www.ibm.com/software/tivoli/education.

Conventions Used in this Book

The following typeface conventions are used in this book:

Bold Lowercase commands or mixed case commands that are difficult to distinguish from surrounding text, keywords, parameters, options, names of Java® classes, and objects are in **bold**.

Italic Variables, titles of publications, and special words or phrases that are emphasized are in *italic*.

`Monospace`
Code examples, command lines, screen output, file and directory names that are difficult to distinguish from surrounding text, system messages, text that the user must type, and values for arguments or command options are in `monospace`.

Chapter 1. Introducing Password Synchronization Plug-ins

This chapter introduces the range of password synchronization plug-ins for IBM Tivoli Directory Integrator. It describes standard concepts, components and procedures.

This chapter contains the following sections:

- "Overview"
- "Building blocks"
- "Available specialized components" on page 4
- "Password Synchronization Architecture and Workflow" on page 5

Note: The Password Synchronization Plug-ins are not included in the IBM Tivoli Directory Integrator 7.1 General Purpose Edition; see *IBM Tivoli Directory Integrator V7.1 Installation and Administrator Guide* for more details.

Overview

The IBM Tivoli Directory Integrator provides an infrastructure and a number of ready-to-use components for implementing solutions that synchronize user passwords in heterogeneous software environments.

A password synchronization solution built with the IBM Tivoli Directory Integrator can intercept password changes on a number of systems. The intercepted changes can be directed back into:

- The same software systems, or
- A different set of software systems.

Synchronization is achieved through the IBM Tivoli Directory Integrator AssemblyLines, which can be configured to propagate the intercepted passwords to desired systems.

Building blocks

The components that make up a password synchronization solution are:

Password Synchronizers
Components which are deployed on the system where password changes occur. They are responsible for intercepting plain (unencrypted) values of the passwords as they are changed.

Java proxy process
Receives passwords from the Password Synchronizers and forwards them to a Password Store.

Password Stores
Components that receive the intercepted passwords, encrypt and store them in locations that can be accessed by the IBM Tivoli Directory Integrator .

Connectors
These are either standard or specialized IBM Tivoli Directory Integrator

Connectors. They connect to locations where the intercepted and encrypted passwords are stored and are able to retrieve and decrypt the passwords.

AssemblyLines
The AssemblyLines use Connectors to get the intercepted passwords and then build custom logic for sending the passwords to other software systems.

Building the solution

The Password Synchronizers, Password Stores and Connectors are ready-to-use components included in the IBM Tivoli Directory Integrator. As a result, implementing the solution that intercepts the passwords and makes them accessible from IBM Tivoli Directory Integrator is achieved by deploying and configuring these components.

For the part of the solution that consolidates passwords intercepted from different sources and feeds these passwords into systems that need to be synchronized, a custom AssemblyLine must be implemented. The look of the AssemblyLine depends mostly on the custom environment and the requirements for the particular solution. IBM Tivoli Directory Integrator does not include these AssemblyLines; they are implemented by the customer.

A password synchronization AssemblyLine usually uses Iterator Connectors to retrieve passwords from the Password Stores. The AssemblyLine then uses other standard Connectors to set these passwords into other systems. If the systems that are synchronized have custom requirements for setting user passwords, these requirements must be addressed in the AssemblyLine and the Connectors that set these passwords. Such customization might consist of setting certain Connector parameters, for example, turning on the **Auto Map AD Password** option in the LDAP Connector to set user passwords in Active Directory. In more complex cases, scripting might be necessary.

A password synchronization solution might include IBM Tivoli Directory Integrator AssemblyLines using Connectors in Server Mode to automate the process of synchronization. For example, an AssemblyLine might listen for changes in the repository where a Password Store component stores the intercepted passwords and trigger the synchronization AssemblyLine whenever a new password is intercepted. Another example might be using an AssemblyLine using a Timer loop that starts the synchronization AssemblyLine on a schedule.

Represented here are some basic and common steps. Each of the components mentioned previously provides interfaces which facilitate the tuning of behavior. Also, the various components can be combined with each other to create custom solutions. These key features provide flexibility for building solutions that meet custom requirements and limitations. The password synchronization suite is mostly comprised of the specialized components that intercept the passwords and make them accessible for the IBM Tivoli Directory Integrator . Once the IBM Tivoli Directory Integrator can access the intercepted passwords through its Connectors, the whole flexibility and openness of the IBM Tivoli Directory Integrator architecture can be leveraged in organizing the process of password retrieval and propagation to other systems.

When NOT to deploy a Password Synchronizer:

There are a number of situations in which it is unwise to use Password Synchronizers, because it would make the solution unneccesarily complex. In particular, in cases involving the Chapter 5, “Sun Directory Server Password Synchronizer,” on page 29 and Chapter 6, “IBM Directory Server Password Synchronizer,” on page 35 simpler methods can be deployed.

As a rule of thumb, Password Synchronizer should be deployed only if hashed password values are not usable outside the directory. Both IBM Directory Server and Sun Directory Server support password encryption where password values are encrypted before they are stored in the directory. Password encryption uses either a one-way or a two-way cryptographic transformation. One-way transformations (for example, hashing with SHA-1 or MD-5) are not reversible. This means that plaintext value cannot be obtained from a one-way encrypted password. The strength of a Password Synchronizer is that it catches the plaintext password before it is hashed and stored in the directory. If hashed values can be used by the destination repository, for example when both the source and the destination systems support the same hashing schemes, synchronization could be done via LDAP and Password Synchronizer is not necessary. As a corollary, you do not need a Password Synchronizer to synchronize passwords between instances of IBM Directory Server and Sun Directory Server. The reason is that both directories support the same set of hashing algorithms for passwords. In those cases you could simply copy passwords between the two instances via LDAP. Alternatively if all you need is to authenticate against an IBM Directory Server with credentials stored in a Sun Directory Server, you can use the pass-through authentication option supported by IBM Directory Server.

Issues with Directory Server replication:

In a replication topology it is recommended to deploy Password Synchronizers on all master instances: When replication is configured, changes are propagated to replication consumers using LDAP operations. If a Password Synchronizer is deployed on a consumer, it will also intercept LDAP operations triggered by replication. If the Password Synchronizer rejects a password coming from replication, the replication process will fail. To avoid such situations, deploy Password Synchronizers on all replication masters, so that passwords could be rejected before they even get into the directory. Be aware that when a password is set on a supplier node in a replication topology, the Password Synchronizers on all associated consumer nodes will synchronize the password value to the Password Store. As a result the same password will be sent to the Store multiple times. One way to avoid this is to configure the directory to use password hashing. Password Synchronizers ignore hashed passwords, so Password Synchronizers on consumers will ignore the already hashed password value, which they receive from the replication supplier.

Hashed Passwords:

The Password Synchronizer ignores hashed password values. This means that only plaintext passwords will be synchronized. The Password Synchronizer would receive hashed passwords in the following cases:

- If an LDAP client sends a password value that is already hashed the directory server will accept it. However, the Password Synchronizer would not be able to obtain a plaintext password from it and will ignore it. For example, if an LDAP client sends "{SHA}5yfRRkrhJDbomacm2lsvEdg4GyY=" instead of "mypass", the Password Synchronizer will not send anything to the Password Store.
- If password encryption is set to one-way transformation (for example, "crypt", "MD5", "SHA-1") passwords are stored in hashed form in the directory.

Consequently replication operations work with hashed password values. This means that Password Synchronizers on replication consumers will receive already hashed password values.

Available specialized components

The following sections describe the specialized password synchronization components that are currently available.

Password Synchronizers

Password Synchronizer for Windows
Intercepts the Windows login password change. See Chapter 4, "Windows Password Synchronizer," on page 17.

Password Synchronizer for Sun Directory Server
Intercepts Sun Directory Server password changes. See Chapter 5, "Sun Directory Server Password Synchronizer," on page 29.

Password Synchronizer for IBM Tivoli Directory Server
Intercepts IBM Tivoli Directory Server password changes. See Chapter 6, "IBM Directory Server Password Synchronizer," on page 35.

Password Synchronizer for Domino®
Intercepts changes of the HTTP password for Lotus® Notes® users. See Chapter 7, "Domino HTTP Password Synchronizer," on page 39.

Password Synchronizer for UNIX and Linux
Intercepts changes of UNIX and Linux user passwords where PAM is enabled. See Chapter 8, "Password Synchronizer for UNIX and Linux," on page 59.

Password Stores

LDAP Password Store
Provides the function necessary to store the intercepted user passwords in LDAP directory servers. See Chapter 9, "LDAP Password Store," on page 63.

JMS Password Store
JMS Password Store (formally known as the MQ Everyplace® Password Store) provides the functionality necessary to store intercepted user passwords in a JMS Provider's Queue from where any JMS client for example, Tivoli Directory Integrator) could read them. See Chapter 10, "JMS Password Store," on page 71.

Log Password Store
The Log Password Store is solely used to log any actions that a normal password store would take. This password store is useful for verifying that the Java Proxy and the native plug-ins are communicating correctly.

Specialized Connectors

JMS Password Store Connector
Provides the function necessary to retrieve password update messages from a JMS Password Store, and send them to IBM Tivoli Directory Integrator. See *IBM Tivoli Directory Integrator V7.1 Reference Guide*, and Chapter 10, "JMS Password Store," on page 71.

Tivoli Identity Manager Integration

This guide also details the steps required for integration between Tivoli Identity Manager and the following Password Synchronizers:

- Sun Directory Server Password Synchronizer,
- IBM Directory Server Password Synchronizer,
- Windows Password Synchronizer, and
- Password Synchronizer for UNIX and Linux.

For more detailed information, see Chapter 13, "Tivoli Identity Manager Integration," on page 83.

Password Synchronization Architecture and Workflow

There are several layers in the IBM Tivoli Directory Integrator Password Synchronizer architecture.

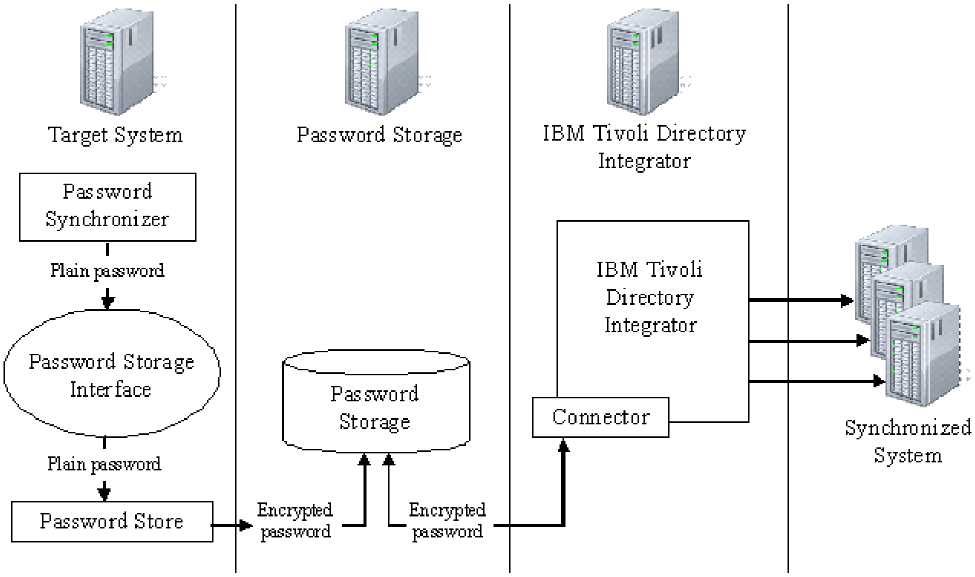

Figure 1. Tivoli Directory Integrator Password Synchronizer architecture

Target System on the diagram designates the software system where we want to intercept password changes. The **Password Synchronizer** component hooks into the Target System using custom interfaces provided by the Target System. The Password Synchronizer component intercepts password changes as they occur in the Target System and before the password is hashed irreversibly.

The Java proxy component is a proxy in the sense that it receives passwords from the server plug-in and redirects them to the Password Storage component. The proxy acts as a container for the Password Storage component – it manages the life-cycle of the Password Storage component and handles inter-process communication with the Tivoli Directory Integrator plug-ins.

The Java Proxy logs any raised errors in the configured log file. If any initialization error is raised the Java Proxy will fail to load. If a runtime error occurs the error is logged for later investigation but the execution of the server continues. This provides high availability in case of a temporary environment change/failure.

Also, a **Password Store** component is deployed on the Target System. Once the Password Synchronizer intercepts a password change it immediately sends the password to the Password Store, using the aforementioned Java proxy process. The Password Store encrypts the password and sends it to a **Password Storage**.

The Password Storage is the second layer in the architecture and represents a persistent storage system (for example, an LDAP directory, or WebSphere® MQ Everyplace) where the intercepted and already-encrypted passwords are stored in a form and location that are accessible from the IBM Tivoli Directory Integrator . The Password Storage can reside on the Target System machine or on another network machine.

The third layer of the architecture is represented by the IBM Tivoli Directory Integrator . The IBM Tivoli Directory Integrator uses a Connector component to connect to the Password Storage and retrieve the passwords stored there. Once in the IBM Tivoli Directory Integrator , the passwords are decrypted and made available to the AssemblyLine that synchronizes them with other systems. The IBM Tivoli Directory Integrator can be deployed on a machine different than the Target System and Password Storage machines.

The next layer in the architecture (in the data flow direction) is represented by the systems whose passwords are synchronized with the Target System. The password synchronization AssemblyLine is responsible for connecting to these systems and updating the passwords there.

The Java Proxy process

The Java proxy component is a proxy in the sense that it receives passwords from the server plug-in and redirects them to the Password Storage component. The proxy acts as a container for the Password Storage component – it manages the life-cycle of the Password Storage component and handles inter-process communication with the TDI plug-ins.

The proxy and the directory plug-in share a common binary command protocol. The communication happens over sockets. The proxy acts as a server, listening for commands. The directory plug-in connects to the proxy, transmits a command and reads the response.

Depending on the configuration the Java Proxy can also do a preliminary validation on the passwords strength. Currently this validation can only be done against the password policies defined in a remote ITIM server. The Java Proxy is responsible for storing password changes received by the plug-in in the configured password store.

Proxy process Authentication

The communication between the various plugins and the Java Proxy is done over sockets. It is restricted to the loopback network interface only. Apart from that a two way authentication takes place each time a connection between the client plugin and the Java Proxy is established. The authentication is based on File System permissions. The authentication procedure uses the *Authentication Folder* (the place where the `pwsync.props` file is situated). It is important to protect the Authentication Folder by means of file system permissions because the authentication process creates one-time-passwords and stores them as files in that folder.

The Authentication Folder need to be secured after the Password Synchronizer is properly setup. In order to do that the folder must be made readable/writable by only the user that executes the process loading the plugin. For example for the Domino HTTP Password Synchronizer the user "notes" is usually the one used to run the Domino Server with. That user must have full control over the Authentication Folder in order for the Password Synchronizer to work.

Note: The Java Proxy needs read/write privileges to the Authentication Folder too. That process is automatically started from the plugin side and thus is executed with the same privileges as the plugin. If for some reason the Java Proxy is started manually by another user then that user need to be granted the read/write access to the Authentication Folder as well. For example if the user *user* has full control over the Authentication Folder you will have to execute the commands with that user's privileges in order for the authentication to succeed.

On Windows:

```
runas /user:user startProxy.bat configuration_file
runas /user:user stopProxy.bat configuration_file
```

On Linux/UNIX:

```
su - user
startProxy.sh configuration_file
stopProxy.sh configuration_file
```

Restricting file access on Windows: The steps outlined in this section will restrict the access to the Authentication Folder for the Windows Password Synchronizer. The plugin is executed in the LSA process owned usually by the local system account. Since that account is part of the Administrators group we need to grant access only to that group by doing the following:

1. Open a file explorer and navigate to the Authentication Folder.
2. Right-click the Authentication Folder and select Properties.
3. On the Security tab click the Advanced button.
4. Deselect the check-box that allows propagation of parent’s permissions and select the check-box that replaces all the child permissions.
5. Remove all the records from the list of Permission Entries.
6. Click the Add button and add the Administrators group. Grant it full control access.
7. Click OK and approve all the warning windows.

If you are setting up a different Password Synchronizer make sure the appropriate user/group is granted the required privileges.

Restricting file access on Linux/UNIX: The steps outlined in this section will restrict the access to the Authentication Folder for the PAM Password Synchronizer. For this purpose *auth_dir* will refer to the folder where the authentication process is taking place.

1. Change the ownership of the folder:
   ```
   chown -R root:root auth_dir
   ```
2. Change the permissions of the folder:
   ```
   chmod -R 700 auth_dir
   ```

The Password Store interface

The Password Store is the place the Java Proxy will store the received passwords.

The Password Store used by a Password Synchronizer can be easily changed when necessary. For example, a Password Synchronizer for IBM Tivoli Directory Server is deployed and configured to use the LDAP Password Store. At some time you decide that you need to use the JMS Password Store. Then you need to configure the JMS Password Store, change a single property of the Password Synchronizer, and restart the IBM Tivoli Directory Server. New password changes are then stored in your designated JMS Password Store, it is not necessary to install the solution again.

Architecture options

For simplicity, the previous diagram shows password interception on a single Target System. Actually, a password synchronization solution might need to intercept password changes on several Target Systems. This is where the layered password synchronization architecture brings additional value in terms of scalability and customization options:

- The Password Store components of several Target Systems can be configured to store the intercepted passwords in the same Password Storage. The IBM Tivoli Directory Integrator AssemblyLine uses a single Connector to connect to the Password Storage and is not affected by the number of Target Systems whose passwords are intercepted and stored in this Password Storage.
- The AssemblyLine can be configured to connect to several Password Storages (using several Iterator Connectors). This is useful when different Password Storages have to be used, or distinction of the Target Systems on IBM Tivoli Directory Integrator is necessary.

In either (or both) of these previous approaches, it is possible to add, remove or change Target Systems in an already existing solution by focusing mainly on the new functionality without affecting the rest of the solution.

On the other end of the data flow, where passwords are updated in systems that you want to keep synchronized, the password synchronization architecture benefits from the inherent scalability of the IBM Tivoli Directory Integrator . Updating passwords on yet another system might be as easy as adding a new Connector in the password synchronization AssemblyLine.

In the case where the Target System is also one of the systems updated with the intercepted passwords from other systems, special care must be taken to avoid circular updates. The implementation on the IBM Tivoli Directory Integrator side must build logic that does not update a system with passwords intercepted on that same system.

Security

Public-private key infrastructure is used to provide secure transport and intermediate storage of password data.

The Password Store components use a public key to encrypt password data before sending it on the wire and storing it in the Password Storage. The IBM Tivoli Directory Integrator AssemblyLine or specialized Connectors have the corresponding private key and use it to decrypt password data retrieved from the Password Storage.

An additional layer of security is added by Password Store components supporting SSL.

The installation folder of each password synchronizer and all files in it must be protected against non-trusted users on the host operating system. The preferred way to achieve this is by setting proper file system permissions – non-trusted users and groups must not have any access (read, write, execute) to the installation folder or the files of the password synchronizer.

Reliability

Functionality for preventing and dealing with possible password de-synchronization is built into the password synchronization workflow.

The Password Synchronizer and Password Store components together provide functionality to deal with cases where an external storage system is not available or malfunctions.

The Password Store always reports to the Password Synchronizer whether or not the password was successfully stored into the Password Storage. The Password Synchronizer component can do the following to prevent or handle possible password de-synchronizations:

- The Password Synchronizer can cancel the password change in the Target System after the Password Store reports that the password is not stored into the Password Storage (due to availability or other reasons), where enabled.
- Where the Target System does not enable cancel or rollback on the password change (which you want to do on unsuccessful storage), the failure is logged with information about the user whose password is not stored in the Password Storage. An Administrator can inspect the log and resolve de-synchronized passwords.

Chapter 2. Installing the Password Synchronization Plug-ins

This chapter describes the installation of the Tivoli Directory Integrator Password Synchronization Plug-ins.

Before You Install

Before you install, please read the following sections and make sure your system meets the minimum requirements.

Platform Requirements

Platform requirements for each Password Synchronization Plug-In is documented in the "Supported Platforms" section of the chapter for each Password Synchronization Plug-In.

Root or Administrator Privileges

On Windows platforms, the installer requires that the user ID used to install IBM Tivoli Directory Integrator Password Synchronization Plug-ins be the Administrator ID or a member of the Administrators group. On UNIX platforms, the installer requires that the user be root. The installer will fail if the user ID used to install IBM Tivoli Directory Integrator Password Synchronization Plug-ins does not have these privileges.

Installing the Password Synchronization Plug-ins

The IBM Tivoli Directory Integrator 7.1 Password Synchronization Plug-ins are installed with the standard Tivoli Directory Integrator Installer; there is no separate installer. See *IBM Tivoli Directory Integrator V7.1 Installation and Administrator Guide* for instruction on how to operate the Installer.

Instead of installing the main product, you must choose a Custom install, and select the Password Synchronization Plug-ins option.

All the Plug-ins will be installed, in a subdirectory of a directory of your choosing, called `install_dir/pwd_plugins`.

The installer will place or alter the following files in the pwd_plugins directory:

domino/pwsync.props
pam/pwsync.props
sun/pwsync.props
tds/pwsync.props
windows/pwsync.props
windows/registerpwsync.reg
windows/unregisterpwsync.reg

Once the Password Synchronization Plug-ins have been laid down by the Installer, then for those Plug-ins you actually want to deploy a number of post-installation steps must be performed. Refer to the actual Plug-ins sections for more information.

Upgrading the Password Synchronization Plug-ins

There is no upgrade functionality of the IBM Tivoli Directory Integrator Password Synchronization Plug-ins Installer. Upgrading from a previous version of the IBM Tivoli Directory Integrator Password Synchronization Plug-ins is achieved by uninstalling the previous version, and installing the new version.

To upgrade Password Synchronization Plug-ins:

1. Make a backup of any file you modified for the previous version.
2. Navigate to the IBM Tivoli Directory Integrator Password Synchronization Plug-ins `_uninst` directory for the previous version. For example:

   install_path/_uninst
3. Launch the uninstaller by executing the uninstall executable.

   Windows

   `uninstall.exe`

   All other platforms

   `uninstall.bin`
4. Reboot the system.
5. Install the new version of Password Synchronization Plug-ins following the instructions given above.
6. Restore the files backed up in step 1, that is, get your customized settings from the backed up configuration files and apply those settings to the new files accordingly.

Chapter 3. Password plug-ins common configuration and utilities

Configuration file parameters

The plug-ins and the Java Proxy share a configuration file, commonly called `pwsync.props`. The path to this file is usually specified when registering the plug-in. The path to the configuration file is then passed to the Java Proxy on startup by the plug-in or by the command line utility that starts the proxy.

Note: The standard `java.util.Properties` class parses the configuration file and replaces control-like characters with actual control characters. This means that when it reads, for example, "\\n", this will be converted to the character '\n'. Therefore when setting a path in that configuration file on the Windows platform the \ character should be escaped with another slash, thus \ would look like this \\.

Common parameters for all Password plug-ins in the configuration file are as follows:

proxyStartExe
: This required string parameter holds the path to an executable (binary or shell script), which will be used to start the Java Proxy. The default value is `*TDI_Install_dir*/pwd_plugins/bin/startProxy.bat(sh)` .

serverPort
: This integer property specifies the port number the Java Proxy will listen to. This property is read by the client plug-in so a connection to the Java Proxy can be established. The default value is 18001.

logFile
: This string parameter configures the log file of the client plug-in. If this parameter is not set no logging will be done.

  Note: The PAM plug-in logs using the UNIX syslog daemon and does not use this property.

checkRepository
: This boolean property enables turning on or off the functionality that checks for availability of the Password Storage.

  When this property is set to true, the Password Synchronizer first checks whether the Password Storage is available. If it is available, the password is changed in the directory, then the password is sent to the Password Storage. If the check indicates that the storage is not available, the LDAP operation (a part of which is the password update) is rejected on the target system.

  When the checkRepository property is set to false, the Password Synchronizer performs no checks for storage availability. The password update is performed in the directory first, then an attempt is made to store it in the Password Storage. If the password cannot be stored, a message is logged in the log file (pointed to by the **logFile** property) to indicate that password synchronization for this user failed.

  The default value is true.

Note: The check for availability of the Password Storage works with all Password Store components.

syncClass
This required property determines which Password Store to use, that is, it configures the class representing the place for storing the passwords.

javaLogFile
This string parameter configures the log file of the Java Proxy. If this parameter is not set no logging will be done for the Java Proxy.

debug This boolean property turns the debugging on or off. Both the client plug-in and the Java Proxy check this property. The default value is false.

Parameters from this configuration file are set as Java system properties. Thus if SSL is required for the communication with any of the stores or with the ITIM servlet the following Java properties must be set in that configuration file:

Table 1. SSL Java Properties

Property	**Value**
javax.net.ssl.trustStore	specifies the trust store for the JVM
javax.net.ssl.trustStorePassword	specifies the password of the trust store **Note:** this should be encrypted using the encryptPasswd utility
javax.net.ssl.trustStoreType	the type of the trust store (usually jks)
javax.net.ssl.keyStore	specifies the key store of the JVM
javax.net.ssl.keyStorePassword	specifies the password for the key store **Note:** this should be encrypted using the encryptPasswd utility
javax.net.ssl.keyStoreType	the type of the key store (usually jks)

Any additional parameters in the configuration file are specific to the actual Password plug-in; see the relevant section for more details.

Command line utilities

The following utilities are available to control certain aspects of the Password Synchronizers configuration and flow process:

TDI_install_dir/**pwd_plugins/bin/encryptPasswd.bat(sh)**
This is the utility used to encrypt passwords before setting them in the various configuration files.

Note: This utility uses a symmetric algorithm to encrypt the passwords. This means that they are easily decrypted by a skilled person. Make sure you allow the reading of the configuration files by trusted users only!

TDI_install_dir/**pwd_plugins/bin/startProxy.bat(sh)**
This is the utility that starts the Java Proxy manually. This utility automatically searches the default jars folder (by default this is `TDI_install_dir/pwd_plugins/jars/`) and creates the classpath of the JavaProxy. For example if you configure the JMS Password Store to work with WebSphere MQ you must add the required MQ jars to the respective plugins' .jar folder so they be automatically loaded next time you start the Java Proxy.

*TDI_install_dir***/pwd_plugins/bin/stopProxy.bat(sh)**

This utility sends a StopRequest to the running Java Proxy process. The result to this is that the Java Proxy will wait until all operations finish and then will exit normally.

When a task calling one of the Password Synchronizers is shut down, the Java proxy process is not automatically terminated. The Password Synchronizer will connect to the proxy process if it is already running, so it is not absolutely necessary to terminate the proxy. However, it is a good practice to do so using the StopProxy utility.

*TDI_install_dir***\pwd_plugins\windows\pwsync_admin.exe**

This utility is used start or stop the Java Proxy and additionally to pause or resume the Windows plug-in. This is the 32-bit version but there is a 64-bit version called pwsync_admin_64.exe in the same folder.

*TDI_install_dir***\jvm\jre\bin\keytool and** *TDI_install_dir***\jvm\jre\bin\ikeyman**

These utilities are used for managing the keystore/trustores used during the plugins setup. Refer to the following topic in the *IBM Tivoli Directory Integrator V7.1 Installation and Administrator Guide* for more details: section Security and TDI -> Secure Socket Layer Support -> Keystore and truststore management.

Refer to the individual Password Synchronizers and Password Stores for instructions as to when to use them.

Manage the Java proxy with IBM Tivoli Monitoring

The Java Proxy process of each Password Synchronizer can be managed by the Agent Management Services of IBM Tivoli Monitoring 6.2.2 FixPack 2. These services are available in the ITM OS Monitoring Agent for Windows, Linux, and UNIX, and are designed keep the Java Proxy process available and to provide information about its status to the Tivoli Enterprise Portal. More information about Agent Management Services can be found at this URL: http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.itm.doc_6.2.2/itm_agentmgmtsvcs_intro.htm.

Managing the Java Proxy with IBM Tivoli Monitoring is entirely optional.

Each Password Synchronizer has an associated CAP file which describes its Java proxy process. After installation all CAP files are located in the `*TDI_install_dir*/pwd_plugins/cap/` directory. The following is a list of the available CAP files:

- Windows Password Synchronizer: tdi_ad_plugin_default.xml
- IBM DS Password Synchronizer: tdi_tds_plugin_default.xml
- Sun DS Password Synchronizer: tdi_sun_plugin_default.xml
- PAM Password Synchronizer: tdi_pam_plugin_default.xml
- Domino HTTP Password Synchronizer: tdi_domino_plugin_default.xml

You must copy the appropriate CAP file to the proper directory for IBM Tivoli Monitoring to recognize that the Proxy is available to be managed.

On UNIX or Linux, this directory is: `/opt/IBM/CAP`

On Windows, this directory is: `%ALLUSERSPROFILE%\ApplicationData\IBM\CAP`

Before you can use the CAP files, you have to modify them to contain the correct path to the installation of Tivoli Directory Integrator.

Chapter 4. Windows Password Synchronizer

This chapter describes the IBM Tivoli Directory Integrator Windows Password Synchronization Plug-in.

This chapter includes the following sections:
- "Overview"
- "Supported Platforms" on page 20
- "Deployment and Configuration" on page 21

Overview

The Password Synchronizer for Windows intercepts password changes of user accounts on Windows operating systems.

Password changes are intercepted in all of the following cases:
- When a user changes his own password through the Windows user interface
- When an administrator changes the password of a user through the Windows administrative user interface
- When a password change request to Active Directory is made through LDAP

The IBM Tivoli Directory Integrator Password Synchronizer plug-in propagates the changes to a repository (Password Store) before the Windows system changes the password.

The IBM Tivoli Directory Integrator Password Synchronizer stores the user password in a Password Store (LDAP server, JMS Password Store).

The change is later propagated to other servers by an IBM Tivoli Directory Integrator AssemblyLine. After the password is successfully stored in the Password Store, control is returned to the Windows system and the user password is modified.

Synchronizing from a single machine

To synchronize passwords from a single machine, install the Password Synchronizer on the Windows standalone machine.

Synchronizing from a Windows XP, Windows 2003 or Windows 2008 domain

To synchronize password changes from a Windows XP, Windows 2003 or Windows 2008 domain, install the Password Synchronizer on all domain controllers for the domain with which you want to synchronize.

Sample scenario

Bob logs onto the windows machine, presses **Ctrl**+**Alt**+**Delete**, and requests a password change. That password change is intercepted by the Password Synchronizer, then delegated to the associated Password Store (LDAP Password Store, JMS Password Store). If the Password Store confirms that the password was successfully stored, then the password change takes place on the native Windows machine, whether it is a standalone machine or a domain controller. If the Password Store indicates that the password was not stored, then the password change on the native Windows machine is denied.

Password change requests to Active Directory through LDAP/JNDI are also intercepted and handled by the Password Synchronizer.

Windows Password Synchronizer Workflow

The Windows Password Synchronizer intercepts a password change before the change is actually committed internally by Windows and Active Directory. The Password Synchronizer passes the new password to the Password Store.

If the Password Store indicates that the password is stored successfully, the Password Synchronizer enables the password change to be committed in Windows.

If the Password Store indicates that the password is not stored, the password change is rejected on the Windows machine. If the password change has been performed from the Windows user interface, an error box is displayed with contents similar to:

```
Windows cannot complete the password change for user_name because:
The password does not meet the password policy requirements.
Check the minimum password length, password complexity and password history
requirements.
```

This is a standard message that is displayed by Windows when the password change is denied. The log files of the Password Synchronizer and the Password Store component indicate the actual reason why the password cannot be stored in the Password Storage.

On each successful password change the Password Synchronizer sends the full name of the user (the "displayName" attribute from Active Directory) to the Password Store. Currently the JMS Password Store ignores this extra data. The LDAP Password Store writes the additional information to the extended data attribute of the user entry (by default the extended data attribute is named "ibm-diExtendedData").

The Password Synchronizer returns the sAMAccountName of the user whose password has been changed. This name is unique for each Windows Domain but it is not unique for the Domain forest. In order to retrieve the rest of the user attributes, additional lookups need to be done using the provided sAMAccountName as link criteria.

Windows Password Synchronizer Filtering

The Windows Password Synchronizer provides filtering functionality. Filtering affects only whether a password change is sent to the password store and not whether the Windows domain accepts or rejects the password change. If the user filter accepts a user, the password changes for that user are sent to the password store. Otherwise, password changes for that user are not sent to the password store for that user.

The user filter makes decisions based on two factors, or criteria:

- group membership,
- DN matching (which is in effect LDAP sub-tree location matching).

Group membership deals with whether a user is a member of some Windows group. The user filter does not recognize nested groups, so if a user is a member of group A, which is nested into group B, then the user will not be deemed member of group B.

DN matching deals with whether a DN suffix matches the Distinguished Name of a user. For example if the user has a Distinguished Name `cn=myuser,ou=myou,dc=mydc,dc=com` it is matched by the DN suffix `dc=mydc,dc=com` but not by `dc=mydc`.

The user filter allows include and exclude rules of both group membership and DN matching. For example the user filter can be configured to accept all users which are members of a certain Windows group (include form) but not members of some other Windows group (exclude form).

Group membership and DN matching in both rules (include/exclude) can be freely combined. However, there is one specific limitation. Exclude rules always have higher priority than include rules. So for example if a user is included by DN matching but excluded by group membership, the user will not be accepted by the user filter.

To preserve backward compatibility, if no include form is specified (neither group membership, nor DN matching), the default form is *include all*. Inversely this also implies that if no exclude form is specified (neither group membership, nor DN matching), the default form is *exclude none*.

Here are some examples to help clarify the filtering mechanism:

- If no configuration is provided to the user filter, it accepts all users (backward compatibility).
- If the user filter is provided with some include rules and no exclude rules, it accepts only users matched by the provided include rules.
- If the user filter is provided with some exclude rules and no include rules, it accepts only users, which are NOT matched by any of the exclude rules.
- If the user filter is provided with some include rules and some exclude rules, it accepts only users, which are matched by some of the include rules and are not matched by any of the exclude rules.

The user filter of the Windows Password Synchronizer is configured using 4 string values in the plug-in configuration file to define the include and exclude rules.

includeGroups
A list of Windows groups. If a user is a member of some group on the list, the user will be accepted by the user filter (assuming the user is not excluded by some of the exclude lists).

excludeGroups
A list of Windows groups. If a user is a member of some group on the list, the user will not be accepted by the user filter.

includeDNs
A list of DN suffixes. If a user's Distinguished Name matches some suffix on the list, the user will be accepted by the user filter (assuming the user is not excluded by some of the exclude lists).

excludeDNs
A list of DN suffixes. If a user's Distinguished Name matches some suffix on the list, the user will not be accepted by the user filter.

All of the above property string values must be lists, with tokens separated by semicolons. Redundant white-spaces are not allowed. The group lists must include only names of existing Windows groups. Matching of a DN suffix against a Distinguished Name is performed by a simple case-insensitive string comparison – no special treatment is provided for white-spaces. For example, the `DC=COM` suffix matches the `cn=myuser,dc=mydc,dc=com` Distinguished Name, but the `dc = com` suffix does not.

If the user filter mechanism encounters an issue (for example an invalid group name in its configuration), an error message is logged and the Windows Password Synchronizer acts as if the filter has accepted the user. If the user filter decides to not accept a user, a message stating that is logged.

The configuration of the user filter is read again on each password notification, so changes to the configuration have immediate effect – there is no need to restart the Windows operating system for the changes to the user filter to be taken into account.

Note: The user filter configuration of the Windows Password Synchronizer is sensitive to modifications of the Windows groups, involved in the configuration. If some of the following changes occur, the Windows Password Synchronizer must be restarted (which requires restart of the operating system):

- the Windows name of a group is modified (this corresponds to the `sAMAccountName` attribute in Active Directory),
- the distinguished name of a group is modified (for example the group is moved to another container).

This restriction holds true for all groups, which have appeared in the configuration of the Windows Password Synchronizer during its lifetime.

Attention: The user filtering feature of the Windows Password Synchronizer will function properly only on machines that are part of a Windows domain. Workgroup machines will not be able to use user filtering. If you configure user filtering on a workgroup machine, the plug-in will log an error message like the following on every password change and will send the change to the password store, no matter the provided configuration:

```
User filtering failed: The specified domain either does not exist or could not be contacted.
```

Note that if no user filtering configuration is supplied, the plug-in will function normally and no error will be logged (because no filtering is performed).

Supported Platforms

The following platforms are supported for the IBM Tivoli Directory Integrator Windows Password Synchronization Plug-in:

- Windows XP Professional (x86)
- Windows 2003 Standard Edition (x86/x86–64)
- Windows 2003 Enterprise Edition (x86/x86–64)
- Windows 2003 Datacenter Edition (x86/x86–64)
- Windows 2008 Standard Edition (x86/x86–64)
- Windows 2008 Enterprise Edition (x86/x86–64)
- Windows 2008 Datacenter Edition (x86/x86–64)

Deployment and Configuration

Post-install configuration

Follow these steps to register the Password Synchronizer for password change notifications:

1. Copy the DLL of the Windows Password Synchronizer to the System32 folder of the Windows installation folder. Note that on 64–bit Windows operating systems, the 64–bit DLL of the Password Synchronizer must be put in the System32 folder.
2. List the name of the Windows Password Synchronizer DLL (without the ".dll" file extension) in the "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\LSA\Notification Packages" Windows registry key (see the following section, “Configuration parameters in the Windows registry”). Make sure you put in the name of the 64–bit DLL on a 64–bit Windows platform.
3. Execute the `registerpwsync.reg` file, which is shipped with the Password Synchronizer. This will create a key for the Windows Password Synchronizer in the Windows registry: "HKEY_LOCAL_MACHINE\SOFTWARE\IBM\Tivoli Directory Integrator\Windows Password Synchronizer". It will also set a string value "ConfigFile" that contains the absolute file name of the configuration file of the Windows Password Synchronizer.

Configuration parameters in the Windows registry

This plugin must be registered in the Windows LSA for receiving password changes notifications. For this purpose the name of the external library must be registered in the specific registry key. Additionally the external library file should be placed in one of the directories that is specified by the PATH environment variable. After this procedure is completed the operating system must be restarted so the external library can be loaded.

Note: If the external library file is registered but could not be loaded successfully for some reason then the Windows OS might become unstable.

When the native module of the Windows Password Synchronizer is initialized, it will read from the registry key folder:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\IBM\Tivoli Directory Integrator\Windows Password Synchronizer]
```

The following registry key is of vital importance, because it contains the location of the configuration file of the Password Synchronizer:

Table 2. Primary registry key

Key name	Type	Description	Required?
ConfigFile	REG_SZ	This key specifies the full path of the configuration file of the Windows Password Synchronizer.	true

Below is a list of optional registry keys which affect the behavior of the Windows Password Synchronizer. You should not set these manually – use the Administration Tool instead.

Table 3. Optional registry keys

Key name	Type	Description	Default	Required?
disabled	REG_SZ	This key specifies whether the password change should be propagated to the Java Proxy process.	false	false
reconfigure	REG_SZ	This key specifies whether the plugin should reload its configuration file on the next password change notification.	false	false

Register the password filter module by editing the key in the following registry key folder:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\LSA
```

The following key should be present:

Table 4. Optional registry keys

Key name	Type	Description	Default	Required?
Notification Packages	REG_MULTI_SZ	This key specifies the external libraries to register for notifications.	unknown	true

Note: Do not delete any of the values of this key. Put the name of the library on the last line. Do not include the .dll extension to the name you enter.

Reboot the Windows machine so that the changes can take effect.

Configuration parameters in the configuration file

The configuration file is named `pwsync.props`. Many of the configuration parameters in this file are common to all Password plug-ins, see "Configuration file parameters" on page 13.

The list below describes only those parameters that are specific to the Windows Password plug-in.

includeGroups
An optional list of Windows groups. If a user is a member of any group in the list, the user will be accepted by the user filter (assuming the user is not excluded by any of the exclude lists).

excludeGroups
An optional list of Windows groups. If a user is a member of any group in the list, the user will not be accepted by the user filter.

includeDNs
An optional list of DN suffixes. If a user's Distinguished Name matches any suffix on the list, the user will be accepted by the user filter (assuming the user is not excluded by any of the exclude lists).

excludeDNs
A list of DN suffixes. If a user's Distinguished Name matches any suffix on the list, the user will not be accepted by the user filter.

accountTypes
This property specifies the type of the account for which password changes will be reported. Its format is a space-delimited list of account types.

The Password Synchronizer plug-in is capable of reporting password changes to the following Windows account types:

NORMAL_ACCOUNT
This is a default account type that represents a typical user.

TEMP_DUPLICATE_ACCOUNT
This is an account for users whose primary account is in another domain.

INTERDOMAIN_TRUST_ACCOUNT
This is a permit to trust account for a domain that trusts other domains.

WORKSTATION_TRUST_ACCOUNT
This is a computer account for a computer that is a member of this domain.

SERVER_TRUST_ACCOUNT
This is a computer account for a backup domain controller that is a member of this domain.

An example value for this key would be:

```
"NORMAL_ACCOUNT WORKSTATION_TRUST_ACCOUNT"
```

Note: The Password Synchronizer always reports password changes to accounts of type NORMAL_ACCOUNT regardless of whether NORMAL_ACCOUNT is specified in the AccountTypes parameter.

Enabling Local Security

Change the Local Security Policy as follows:

1. Select **Control Panel**>**Administrative Tools**>**Local Security Policy**
2. Select **Account Policies**>**Password Policy**
3. Change **Passwords must meet complexity requirements** to **enabled**.

Notes:

1. For this change to take place, reboot the machine. Make sure that you set up the Password Store properties file before rebooting the machine.
2. If the Windows Server is configured as a Domain Controller, the "Passwords must meet complexity requirements" setting needs to apply to the whole Active Directory Domain, therefore this setting should be modified using the "Domain Security Policy" tool.

Password Stores setup information

The installer will configure the Password Synchronizer to use the Log Password Store by default.

For information on setting up the Password Stores, see the following resources:

- Chapter 9, “LDAP Password Store,” on page 63.
- Chapter 10, “JMS Password Store,” on page 71.
- Chapter 11, “Log Password Store,” on page 79.

Plug-in administration tool

A command line tool for performing administrative tasks, `pwsync_admin.exe`, can be found in the plug-in installation directory. The primary purpose of this administrative tool is to allow reconfiguration of the Windows Password Synchronizer without rebooting the Windows machine. For example, this tool enables changing of the password store without rebooting Windows.

Note: The only change that cannot be accomplished without rebooting Windows is replacing the `tdipwflt.dll` plug-in, located in the Windows `System32` directory.

Usage

This is how the administration tool is used from the command line:

```
pwsync_admin.exe – command for 32 bit Windows
pwsync_admin_64.exe – command for 64 bit Windows
```

This tool takes a single command line parameter (the command argument above), which can have one of the following values:

suspend_plugin
This command writes a boolean value to the Windows registry (please see the Windows registry settings section), thus indicating to the plug-in that subsequent password changes must not be propagated to the Java proxy. This command causes subsequent password changes to be skipped until a **resume_plugin** command is issued.

resume_plugin
This command writes a boolean value to the Windows registry (please see the Windows registry settings section), thus indicating to the plug-in that subsequent password changes must be propagated to the Java proxy. This command causes subsequent password changes to be synchronized until a **suspend_plugin** command is issued.

reconf_plugin
This command writes a boolean value to the Windows registry (please see the Windows registry settings section), thus indicating that the plugin must reload its configuration file. Reloading will not happen immediately but rather on the next password change. This means that if there are any errors with the new configuration, they will not become evident immediately. You could trigger a password change of a test account to enforce the reconfiguration. Beware that reconfiguration will be postponed if the plugin is suspended.

query_plugin
This command queries the status of the plugin – whether the plugin is currently loaded and if its last initialization was successful.

stop_proxy
This command causes the administration tool to connect through a socket to the command socket port of the Java proxy and send a stop request to the proxy. This causes the proxy to terminate gracefully.

start_proxy
This command starts the Java proxy, which causes the proxy configuration to be reloaded.

restart_proxy
This command is equivalent to a **stop_proxy** command followed by a **start_proxy** command.

query_proxy
command determines whether the Java Proxy is running or not.

Operational Windows registry settings

There are a number of Windows registry keys associated with the Windows Password Plug-in and its operations:

Enable or disable plugin

The registry key used by the suspend_plugin and resume_plugin commands is:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\IBM\Tivoli Directory Integrator\Windows Password Synchronizer] "disabled"="true"
```

If the key has a value of true, then the plug-in will not synchronize passwords. If this key is missing or has a value other than true, the plug-in will synchronize passwords. This key is created by the plug-in administration tool on first use.

Reload plugin configuration

The reconf_plugin command uses the following registry key:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\IBM\Tivoli Directory Integrator\Windows Password Synchronizer] "reconfigure"="true"
```

If the key is set to true, then on the next password change the plugin will reload its configuration file. The plugin will also change the value to "false", so that the reload happens only once.

Note: Neither of the above keys is present in the Windows registry after the plug-in is installed. These keys are not required for the normal operation of the plug-in.

Logging

The administrative tool logs messages both to the console and to a log file named `pwsync_admin.log`, which is located in the install directory of the plug-in. The log file can be used for analyzing errors encountered during administrative tool operations, or an historical reference for operations performed using this tool.

Considerations when using the administration tool

When using the administration tool, be aware of the following considerations:

- When the plug-in is suspended, password changes are skipped (not propagated) by the plug-in. This can result in inconsistencies (password changes lost) in the target synchronization system
- The plug-in will attempt to restart the Java proxy only if reconfiguration is requested (see the "reconf_plugin" admin tool command) and the proxy is not already running.
- When the Java proxy is started, it loads the password store configuration file. This happens when the machine is rebooted, or when the plug-in is not suspended but the Java proxy is stopped as a password change occurs. If the user is editing the configuration file at the time, the Java proxy may load a possibly corrupted configuration.
- When the plug-in is not suspended and the Java proxy is not running, if a password change is issued with the **Active Directory Users and Computers** user interface tool, the plug-in is notified by Windows two or three times of this password change. The result is that the same password update is propagated two or three times. This happens because the plug-in starts the proxy on the next password change, which takes some time. This causes Windows to notify the plug-in several times of the same password change. This multiple reporting, however, is only present the first time the Java proxy is not running, because on subsequent password changes the Java proxy is already running.
- When the plug-in is configured with the LDAP Password Store and the LDAP Store itself is set for asynchronous storing (`waitForStore=false` specified in the

LDAP Store configuration file), and when the plug-in is not suspended, it is possible that a **stop_proxy** command would cause some password changes to be skipped.

The following recommendations help address these problems:

- Suspend the plug-in using a **suspend_plugin** command prior to any **stop_proxy** or **restart_proxy** commands.
- Make a copy of the configuration file for editing purposes. Replace the old configuration file with the new one when all edits are complete.
- Make any necessary configuration changes at a low usage time, so that few (if any) password changes will be skipped and not propagated.

Example for changing the configuration without rebooting the Windows machine

The following steps show how the configuration settings can be changed without rebooting the Windows machine:

Note: After these steps are completed the plugin, the Java proxy and the password store will use the new configuration settings. During the short window when the plug-in is suspended, however, password changes could be skipped. They will occur in the Windows domain controller, but they will not be propagated by the plug-in. Therefore, this procedure should occur at a low usage time, when password changes are unlikely.

1. Copy the configuration file to a temporary location.
2. Edit the file in this temporary location.
3. Copy the edited file back to the original location.
4. Run the **pwsync_admin.exe suspend_plugin** command.
5. Run the **pwsync_admin.exe reconf_plugin** command
6. Run the **pwsync_admin.exe stop_proxy** command.
7. Run the **pwsync_admin.exe start_proxy** command.
8. Run the **pwsync_admin.exe resume_plugin** command.

Alternatively, if you wish to change only some Password Store settings (and not settings related to the plugin or the proxy) you may skip the reconfiguration command in the above steps.

Migration from previous installations

There is no automatic migration from previous versions. You can migrate existing configurations following the steps below. Note that you should keep the existing configurations before uninstalling the older version:

- Record the settings under the Windows registry folder of the Windows Password Synchronizer. That registry folder will look like one of the following (refer to the documentation for the exact version of the Windows Password Synchronizer):

  ```
  [HKEY_LOCAL_MACHINE\SOFTWARE\IBM\Tivoli Directory Integrator 6.1\Windows Password Synchronizer]
  [HKEY_LOCAL_MACHINE\SOFTWARE\IBM\Tivoli Directory Integrator 6.1.1\Windows Password Synchronizer]
  ```

- Keep the configuration file of the Password Store (for example, "mqepwstore.props" or "idipwstore.props").
- If you are using the MQe Password Store you may want to keep the configuration file that you used to create to generate the files for the MQe Broker (for example, mqeconfig.props) with the mqeconfig utility.

After you have stored the above configurations in a safe place, you must uninstall the old version of the Windows Password Synchronizer and install version 7.0 or later.

Follow the steps below to configure the new installation:

1. Execute the post-install configuration steps.
2. In the configuration file (pwsync.props), set property "syncClass" to an appropriate value, corresponding to the value of the "Class" registry key from the previous installation. (see “Available specialized components” on page 4 for class names of available Password Stores in IBM Tivoli Directory Integrator 7.1).
3. In the configuration file, set property "serverPort" to the value of the "ProxyCommandPort" registry key from the previous installation.
4. In the configuration file, set properties "accountTypes", "includeGroups", "includeDNs", "excludeGroups" and "excludeDNs" to the values of the registry keys with the same names from the previous installation.
5. If you were using an MQe Password Store, use the `mqeconfig` utility and the `mqeconfig.props` file from the previous version to create the files for the MQe Broker.
6. Migrate the Password Store configuration from the previous version (`mqepwstore.props` or `idipwstore.props`) as described in the sections for the respective Password Stores (Chapter 9, “LDAP Password Store,” on page 63 and Chapter 10, “JMS Password Store,” on page 71.)

Reliability and availability

Initialization failure:

If the Password Synchronizer fails to initialize itself (for example, because it cannot find its configuration file), Windows will not send notifications about password changes to the Password Synchronizer. This means that password changes can take place, but the Windows Password Synchronizer will not intercept them.

The most reliable way to determine whether the Password Synchronizer is initialized successfully is to check its error log. Additionally, you can use the `query_plugin` command of the Plugin Administration Tool.

Reconfiguration failure:

If reconfiguration fails, the Password Synchronizer will be left in a non-initialized state and will reject all password changes. Inspect the error log of the Password Synchronizer to see if the reconfiguration succeeded.

See “Plug-in administration tool” on page 23 for information how to request reconfiguration.

Chapter 5. Sun Directory Server Password Synchronizer

This chapter describes the Tivoli Directory Integrator Sun Directory Server Password Synchronization Plug-in.

This chapter includes the following sections:

- "Overview"
- "Supported platforms" on page 30
- "Deployment and configuration" on page 30

Overview

The Sun Directory Server Password Synchronizer intercepts changes to LDAP passwords in Sun Directory Server.

In many cases, it may be possible to build a solution that synchronizes passwords, but without using this plug-in; see "Building the solution" on page 2 for more information.

The Sun Directory Password Synchronizer consists of the following parts:

Sun Directory Server plug-in
The plug-in is a native binary, which uses the Plug-in API of the Sun Directory Server. It runs in the process of the Sun Directory Server.

Java proxy
This is a separate Java process, which is launched/stopped by the server plug-in. Its main purpose is to host the Password Storage component and communicate with the plug-in part. For more information on the Java Proxy, see "Password Synchronization Architecture and Workflow" on page 5.

Password Storage component
This is a Java component, which runs inside the process of the Java proxy and puts passwords into a particular Password Store (LDAP directory, message queue). For more information on Password Storage components see "Available specialized components" on page 4.

Passwords in Sun Directory Server are stored in the `userPassword` LDAP attribute. The Password Synchronizer intercepts updates of the `userPassword` LDAP attribute.

The Sun Directory Server Password Synchronizer intercepts modifications of the `userPassword` attribute of entries of any object class.

Password updates are intercepted for the following types of entry modifications:

- When a new entry is added in the directory and the entry contains the `userPassword` attribute.
- When an existing entry is modified and one of the modified attributes is the `userPassword` attribute. This includes the following cases:
  - The `userPassword` attribute is added (for example, the entry did not have a `userPassword` attribute before)

- The userPassword attribute is modified (for example, the entry had this attribute and its value is now changed)
- The userPassword attribute is deleted from the entry

Notes:

1. Deletion of entries is not intercepted by the Sun Directory Server Password Synchronizer even when the entry contains the userPassword attribute.
2. The userPassword attribute in Sun Directory Server is multiple-valued. Users might have several passwords. The Sun Directory Server Password Synchronizer intercepts and reports any change of any of the password values.

Hashed Passwords:

The Password Synchronizer ignores hashed password values. This means that only plaintext passwords will be synchronized. The Password Synchronizer receives hashed passwords in the following cases:

- If an LDAP client sends a password value which is already hashed, the directory server will accept it. However, the Password Synchronizer would not be able to obtain a plaintext password from it and will ignore it. For example, if an LDAP client sends "{SHA}5yfRRkrhJDbomacm2lsvEdg4GyY=" instead of "mypass", the Password Synchronizer will not send anything to the Password Store.
- If password encryption is set to one-way transformation (for example, "crypt", "MD5", "SHA-1") passwords are stored in hashed form in the directory. Consequently replication operations work with hashed password values. This means that Password Synchronizers on replication consumers will receive already hashed password values.

Supported platforms

The Sun Directory Server Password Synchronizer is available for the Sun Directory Server on the following platforms:

- Windows 2003 Standard Edition (x86), Sun ONE 5.2, Sun Java System Directory Server 6.0 (32-bit)
- Windows 2003 Enterprise Edition (x86), Sun ONE 5.2, Sun Java System Directory Server 6.0 (32-bit)
- Windows 2003 Datacenter Edition (x86), Sun ONE 5.2, Sun Java System Directory Server 6.0 (32-bit)
- Solaris 9 SPARC (32/64-bit), Sun ONE 5.2, Sun Java System Directory Server 6.0 (32/64-bit)
- Solaris 10 SPARC (32/64-bit), Sun ONE 5.2, Sun Java System Directory Server 6.0 (32/64-bit)

Deployment and configuration

This section describes the steps required to install the plug-in on a Windows, UNIX or Linux platform.

Registering the Sun Directory Server Password Synchronization Plug-ins with Sun Directory Server

Sun ONE Directory Server 5.2

To register the plug-in, stop Sun Directory Server and add the following to the Sun Directory Server configuration file `dse.ldif`, using the Directory Server Management Console:

```
dn: cn=IBM DI PassSync,cn=plugins,cn=config
nsslapd-pluginPath: TDI_install_dir/pwd_plugins/sun/sunpwsync.dll
nsslapd-pluginEnabled: on
objectClass: top
objectClass: nsSlapdPlugin
objectClass: extensibleObject
cn: IBM DI PassSync
nsslapd-pluginType: object
nsslapd-pluginInitfunc: PWSyncInit
nsslapd-pluginarg0: TDI_install_dir/pwd_plugins/sun/pwsync.props
nsslapd-pluginId: ibmdi.pwsync
nsslapd-pluginVersion: 7.1
nsslapd-pluginVendor: IBM
nsslapd-pluginDescription: IBM Tivoli Directory Integrator plug-in for password synchronization
```

Note: According to the SUN Directory Server Documentation, the 64-bit Sun DS server running on Solaris will search the 64-bit libraries in a directory under the specified path.

For example, if the value of nsslapd-pluginPath is set in the configuration entry as follows:

```
nsslapd-pluginPath: TDI_install_dir/pwd_plugins/sun/libsunpwsync_64.so
```

then a 64-bit Directory Server running in Solaris Operating Environment searches for a 64-bit plug-in library named: `*TDI_install_dir*/pwd_plugins/sun/64/libsunpwsync_64.so`

That is why on Solaris the 64-bit binary for the Sun Directory Server Password Synchronizer is shipped in that folder instead.

Note: Generally you should avoid manually modifying the `dse.ldif` configuration file of the Sun Directory Server. You can register the plug-in by importing the above LDIF statements using the Directory Server Console like this:

1. Save the above LDIF content in an LDIF file (replace the placeholders for the installation folder accordingly).
2. Open the Directory Server instance in the Directory Server Console.
3. Go to the **Tasks** tab.
4. Select **Import LDIF**.
5. Browse to the location of the file. Check the **Add only** checkbox. Uncheck the **Continue on error** checkbox. Click **OK**.
6. Restart the Directory Server, so that the plug-in is loaded

Sun Java System Directory Server Enterprise Edition 6.0

Locate the `dsconf` command-line tool that ships with the Directory Server; it will be used to register the plug-in. Ensure the Directory Server is running. Execute the following steps (refer to the notes after the steps for the meaning of the *access-options* placeholder):

1. Register the plug-in binary (you may need to change the name of the binary depending on the platform – sunpwsync.dll, libsunpwsync.so, and so forth):

```
dsconf create-plugin <access options> -H "TDI_install_dir/pwd_plugins/sun/sunpwsync.dll"
 -F PWSyncInit -Y object -G "TDI_install_dir/pwd_plugins/sun/pwsync.props" "IBM DI PassSync"
```

2. Set the description of the plug-in:

   ```
   dsconf set-plugin-prop access-options "IBM DI PassSync"
    "desc:IBM Tivoli Directory Integrator plug-in for password synchronization"
   ```
3. Set the vendor name of the plug-in:

   ```
   dsconf set-plugin-prop access-options "IBM DI PassSync" vendor:IBM
   ```
4. Set the version of the plug-in:

   ```
   dsconf set-plugin-prop access-options "IBM DI PassSync" version:7.1
   ```
5. Enable the plug-in:

   ```
   dsconf enable-plugin access-options "IBM DI PassSync"
   ```
6. Restart the Directory Server, so that it loads the plug-in.

Notes:

1. The *access-options* placeholder should be replaced with access details and credentials used to connect to the Directory Server.

   For example, if the Directory Server is located on the localhost, accepts non-SSL connections on port 1389 and the uses the default administrator DN *cn=Directory Manager*, you can use the following options:

   ```
    –p 1389 ––unsecured
   ```

   For a full list of options that `dsconf` supports refer to the Sun documentation: http://docs.sun.com/app/docs/doc/819-0986/6n3chglmh?a=view.
2. To un-register the plug-in use the following command:

   ```
   dsconf delete-plugin access-options "IBM DI PassSync"
   ```

Enable Sun Directory Server logging for plug-ins

The directory plug-in part of the Sun Directory Server Password Synchronizer logs messages in the error log of the Sun Directory Server. By default messages from server plug-ins do not appear in the error log for performance reasons.

Sun ONE Directory Server 5.2

Perform the following steps to enable Sun Directory Server logging for plug-ins:

1. On the Directory Server Console, select the **Configuration** tab.
2. In the navigation tree, expand the Logs folder and select the Error Log icon.
3. The error log configuration attributes are displayed in the pane on the right side of the screen.
4. To enable error logging, select the **Enable Logging** check box (error logging is enabled by default).
5. Select **Plug-ins** in the Log Level list box and click **Save**.

Sun Java System Directory Server Enterprise Edition 6.0

Perform the following steps to enable Sun Directory Server logging for plug-ins:

1. Ensure the Directory Server is running.
2. Execute the following command with the `dsconf` tool of the Directory Server:

   ```
   dsconf set-log-prop access-options error level:err-plugins
   ```

   For the meaning of the *access-options* placeholder see the notes in section "Sun Java System Directory Server Enterprise Edition 6.0" on page 31.

To query the current level of the error log run the following command:

```
dsconf get-log-prop access-options error level
```

Configuring the Sun Directory Server Password Synchronization Plug-in

The Sun Directory Server plug-in has a template configuration file installed at *TDI_Install_dir*/pwd_plugins/sun/pwsync.props. When the SunDS plug-in is initialized, it will expect that the configuration file is set as the last parameter of the plug-in's registration line. The plug-in then reads the file. Some of the parameters in that configuration file are shared between the plug-in and the Java Proxy. For a complete list of the supported properties, check out Chapter 3, “Password plug-ins common configuration and utilities,” on page 13.

The property listed below is specific for the Sun Directory Server Password Synchronizer:

syncBase

This optional property enables restricting the part of the directory tree where passwords are intercepted. The string value specified is the LDAP distinguished name (dn) of the root of the tree whose entry' passwords you want to intercept. Specifying "o=ibm,c=us", for example, results in intercepting password update "cn=Kyle Nguyen,ou=Austin,o=IBM,c=US" and skipping the password update "cn=Henry Nguyen,o=SomeOtherCompany,c=US". Setting no value for this property results in the interception of password updates in the whole directory tree.

Chapter 6. IBM Directory Server Password Synchronizer

The chapter describes the configuration and operation of the IBM Tivoli Directory Integrator Directory Server Password Synchronizer.

This chapter contains the following sections:

- "Overview"
- "Supported platforms" on page 36
- "Deployment and configuration" on page 36

Overview

The IBM Tivoli Directory Server Password Synchronizer intercepts changes to LDAP passwords in IBM Tivoli Directory Server.

In many cases, it may be possible to build a solution synchronizing passwords, but without using this plug-in; see "Building the solution" on page 2 for more information.

The IBM Directory Password Synchronizer consists of the following parts:

IBM Directory Server plug-in
The plug-in is a native binary, which uses the Plug-in API of the IBM Directory Server. It runs in the process of the IBM Directory Server.

Java proxy
This is a separate Java process, which is launched/stopped by the server plug-in. Its main purpose is to host the Password Storage component and communicate with the plug-in part. For more information on the Java Proxy, see "Password Synchronization Architecture and Workflow" on page 5.

Password Storage component
This is a Java component, which runs inside the process of the Java proxy and puts passwords into a particular Password Store (LDAP directory, message queue). For more information on Password Storage components see "Available specialized components" on page 4.

Passwords in IBM Tivoli Directory Server are stored in the `userPassword` LDAP attribute. The Password Synchronizer intercepts updates of the `userPassword` LDAP attribute.

The IBM Tivoli Directory Server Password Synchronizer intercepts modifications of the `userPassword` attribute of entries of any object class.

Password updates are intercepted for the following types of entry modifications:

- When a new entry is added in the directory and the entry contains the `userPassword` attribute.
- When an existing entry is modified and one of the modified attributes is the `userPassword` attribute. This includes the following cases:
  - The `userPassword` attribute is added (for example, the entry did not have a `userPassword` attribute before)

- The `userPassword` attribute is modified (for example, the entry had this attribute and its value is now changed)
- The `userPassword` attribute is deleted from the entry

Notes:

1. Deletion of entries (users) is not intercepted by the IBM Tivoli Directory Server Password Synchronizer even when the entry contains the `userPassword` attribute.
2. The `userPassword` attribute in IBM Tivoli Directory Server is multiple-valued. Users can have several passwords. The IBM Tivoli Directory Server Password Synchronizer intercepts and reports any change of any of the password values.

Supported platforms

The IBM Tivoli Directory Server Password Synchronizer is available for the IBM Tivoli Directory Server on the following platforms and for the following versions:

- Windows 2003 Standard Edition (x86/x86–64), TDS 6.1, TDS 6.2 (32/64–bit)
- Windows 2003 Enterprise Edition (x86/x86–64), TDS 6.1, TDS 6.2 (32/64–bit)
- Windows 2003 Datacenter Edition (x86/x86–64), TDS 6.1, TDS 6.2 (32/64–bit)
- Windows 2008 Standard Edition (x86/x86–64), TDS 6.1, TDS 6.2 (32/64–bit)
- Windows 2008 Enterprise Edition (x86/x86–64), TDS 6.1, TDS 6.2 (32/64–bit)
- Windows 2008 Datacenter Edition (x86/x86–64), TDS 6.1, TDS 6.2 (32/64–bit)
- AIX 5L 5.3 (5300-03) (64 bit), TDS 6.1, TDS 6.2 (64–bit)
- AIX 6.1 (64 bit), IDS 6.0 (64 bit), TDS 6.1, TDS 6.2 (64–bit)
- Solaris 9 SPARC (64–bit), TDS 6.1, TDS 6.2 (64–bit)
- Solaris 10 SPARC (64–bit), TDS 6.1, TDS 6.2 (64–bit)
- RHEL ES/AS 4.0 (x86/x86–64), TDS 6.1, TDS 6.2 (32/64–bit)
- RHEL ES/AS 5.0 (x86/x86–64), TDS 6.1, TDS 6.2 (32/64–bit)
- SLES 9 (x86/x86–64), TDS 6.1, TDS 6.2 (32/64–bit)
- SLES 10 (x86/x86–64), TDS 6.1, TDS 6.2 (32/64–bit)
- RedFlag Data Center 5.0 SP1/Asianix 2.0 SP1, TDS 6.1, TDS 6.2 (32–bit)

Deployment and configuration

Register the plug-in with the IBM Directory Server

To register the plug-in, edit the IBM Directory Server configuration file `*ids_dir*/etc/ibmslapd.conf`.

Note: Before editing the file, make sure the server is not running.

Find the section `dn: cn=Directory, cn=RDBM Backends, cn=IBM Directory, cn=Schemas, cn=Configuration` and add the following (as one line):

Win32 ibm-slapdPlugin: preoperation "*TDI_Install_dir*\pwd_plugins\tds\idspwsync.dll" PWSyncInit "*TDI_Install_dir*\pwd_plugins\tds\pwsync.props"

AIX64 ibm-slapdPlugin: preoperation "*TDI_Install_dir*/pwd_plugins/tds/libidspwsync_64.a.so "PWSyncInit "*TDI_Install_dir*/pwd_plugins/tds/pwsync.props"

Linux32

ibm-slapdPlugin: preoperation "*TDI_Install_dir*/pwd_plugins/tds/libidspwsync.so" PWSyncInit "*TDI_Install_dir*/pwd_plugins/tds/pwsync.props"

Start the IBM Directory Server again.

Configuring the IBM Directory Server Password Synchronization Plug-in

The IBM Directory Server plug-in has a template configuration file installed at `TDI_Install_dir/pwd_plugins/sun/pwsync.props`. When the TDS plug-in is initialized, it will expect that the configuration file is set as the last parameter of the plug-in's registration line. The plug-in then reads the file. Some of the parameters in that configuration file are shared between the plug-in and the Java Proxy. For a complete list of the supported properties, check out Chapter 3, "Password plug-ins common configuration and utilities," on page 13.

The property listed below is specific for the IBM Directory Server Password Synchronizer:

syncBase

This optional property enables restricting the part of the directory tree where passwords are intercepted. The string value specified is the LDAP distinguished name (dn) of the root of the tree whose entry' passwords you want to intercept. Specifying "o=ibm,c=us", for example, results in intercepting password update "cn=Kyle Nguyen,ou=Austin,o=IBM,c=US" and skipping the password update "cn=Henry Nguyen,o=SomeOtherCompany,c=US". Setting no value for this property results in the interception of password updates in the whole directory tree.

Chapter 7. Domino HTTP Password Synchronizer

This chapter describes the configuration and operation of the IBM Tivoli Directory Integrator Domino HTTP Password Synchronizer.

This chapter contains the following sections:

- “Overview”
- “Supported platforms”
- “Installation” on page 40
- “Configuration file options” on page 40
- “Deployment on Domino servers” on page 43
- “Using the Password Synchronizer” on page 55

Overview

The Domino HTTP Password Synchronizer intercepts changes of the Internet password (also known as HTTP password) for Notes users.

The following types of password changes are intercepted:

Administrative password resets
A user with the necessary rights (usually an administrator) changes his or another user's password without being prompted for the old password:

- The HTTP password is changed by editing the **Internet password** field of the user's Person document using the Lotus Domino Administrator client.
- The HTTP password is changed by editing the **Internet password** field of the user's Person document using the Web browser interface.

Normal user password changes
A user changes his own password and is prompted for the old password:

- A user changes his password from a Web browser using the **Change Password** form from the `domcfg.nsf` (Domino Web Server Configuration) database.
- A user changes his password from iNotes®.

Supported platforms

The Domino HTTP Password Synchronizer is supported on the following platforms:

- Windows Server 2003 Standard Edition (x86/x86-64), Domino 8.0 and Domino 8.5
- Windows Server 2003 Enterprise Edition (x86/x86-64), Domino 8.0, and Domino 8.5
- Windows Server 2003 Datacenter Edition (x86/x86-64), Domino 8.0, and Domino 8.5
- Windows Server 2008 Standard Edition (x86/x86-64), Domino 8.0 and Domino 8.5
- Windows Server 2008 Enterprise Edition (x86/x86-64), Domino 8.0, and Domino 8.5

- Windows Server 2008 Datacenter Edition (x86/x86-64), Domino 8.0, and Domino 8.5
- AIX® 5L 5.3 (5300-03) (32/64 bit), Domino 8.0 and Domino 8.5
- AIX 6.1 (32/64 bit), Domino 8.0 and Domino 8.5
- Solaris 10 SPARC (32/64 bit), Domino 8.0 and Domino 8.5
- SLES 9 (x86), Domino 8.0
- SLES 9 (x86–64), Domino 8.0 and Domino 8.5
- SLES 10 (x86), Domino 8.0 and Domino 8.5
- SLES 10 (x86–64), Domino 8.0 and Domino 8.5
- RHEL ES/AS 4.0 (x86), Domino 8.0
- RHEL ES/AS 4.0 (x86–64), Domino 8.0 and Domino 8.5
- RHEL ES/AS 5.0 (x86), Domino 8.0 and Domino 8.5
- RHEL ES/AS 5.0 (x86–64), Domino 8.0 and Domino 8.5
- Red Flag Data Center 5.0 SP1 /Asianix 2.0 SP1, Domino 8.0 and Domino 8.5

Installation

The Domino HTTP Password Synchronizer is installed using the standard Tivoli Directory Integrator installer wizard.

Read the section "Configuration file options" for available configuration properties. Configure both the plug-in and the Java Proxy.

Read the section "Post install configuration" on page 41 for information about the required intermediate installation steps.

Read about the various password stores in this Guide (like the Chapter 9, "LDAP Password Store," on page 63) and configure one.

Register the plug-in to the Domino Server as described in the section "Deployment on Domino servers" on page 43.

Configuration file options

The Domino plug-in has a template configuration file installed at *TDI_Install_dir*/pwd_plugins/domino/pwsync.props. When the Domino plug-in is initialized, it will expect that the configuration file is placed at *domino_data_dir*/idipwsync/pwsync.props on UNIX or at *domino_program_dir*\idipwsync\pwsync.props on Windows. The plug-in then reads the file. Some of the parameters in that configuration file are shared between the plug-in and the Java Proxy. For a complete list of the supported properties, see Chapter 3, "Password plug-ins common configuration and utilities," on page 13.

The Domino Password Synchronizer supplies an option to synchronize password changes using a unique User Identifier. The User Identifier, supplied by Domino, uniquely identifies users within their corresponding Domino Server.

The following common properties are ignored by the Domino plug-in:

proxyStartExe

The Java Proxy is started as a Domino Task when the Domino Server is started. The Java Proxy is automatically stopped when the Domino Server is shutting down; it can be manually shut down using the stopProxy script.. To start the Java Proxy, the Domino Server instantiates the class com.ibm.di.plugin.domino.ProxyLoader, which is a native Domino Addin representing a separate Domino Task. The Domino Server is configured to start another JVM dedicated for that Domino Task. The configuration is done when editing the notes.ini file and adding the "runjava" line.

logFile
This property is ignored because the Domino plug-in uses three log files instead of one. See the section below to find out which properties configure those files.

In addition to the common configuration properties, the Domino plug-in recognizes the following properties:

admin.logFile
This property sets the file the admin agent will log into. If this file is not set the Agent will not output any log. The default value is `idipwsync/admin.log`.

client.logFile
This property sets the file the client agent will log into. If this file is not set the Agent will not output any log. The default value is `idipwsync/client.log`.

web.logFile
This property sets the file the web agent will log into. If this file is not set the Agent will not output any log. The default value is `idipwsync/web.log`.

useUniqueID
This boolean property turns the usage of the unique user ID on or off. If this property is set to true then the plug-in will send the unique id instead of the actual user-name. The default value is false.

ignoreMissingUniqueID
If this boolean property is set to true and the useUniqueID is also true, then the plug-in will skip the synchronization for users for which UNID could not be found. The default value is false.

usernamePrefix
If this property is set to true and the useUniqueID is also true and the ignoreMissingUniqueID is set to false, then the plug-in will prefix the user's distinguished name with the value of this property. The default value is "" (empty string).

Post install configuration

Copy all the files from the folder *TDI_Install_dir*`/pwd_plugins/jars` to the folder *domino_jvm_directory*`/lib/ext` folder on the Domino Server and to the `Lotus\Notes\jvm\lib\ext` folder on the machine where Lotus Domino Designer is installed.

Copy the files `idipwsync.nsf` and `pwsync_install_r8.nsf` from the folder *TDI_Install_dir*`/pwd_plugins/domino` to the data directory of the Domino Server: *domino_data_dir*.

Copy the Domino/JavaProxy configuration file (template is shipped in *TDI_Install_dir*`/pws_plugins/domino/pwsync.props`) to the file *domino_data_directory*`/idipwsync/pwsync.props` for Unix or to the *domino_program_directory*`\idipwsync\pwsync.props` for Windows.

Note: On Linux- and UNIX-based platforms, install the Password Store with the Domino user (`notes` by default). This gives the necessary privileges to the Domino JVM to execute the Password Store. Also make sure the Domino user has the necessary privileges to read the files copied to the Domino Server (those described above).

In order for the new files to be loaded a restart of the server is required.

Creating a signer for the Password Synchronizer agents

Here are sample steps to create a person that can be used as agent signer:

- Open the Domino Administrator.
- Open the **People & Groups** page.
- On the right panel select **People -> Register...** (the Register Person wizard will show up).
- In the Last name field enter "IDIPWSyncSigner".
- Fill in the password field with a value of your choice.
- In the Mail system field select **None**.
- Make sure **Create a Notes ID for this person** is checked (the ID file will be used to sign agents).
- Click the **Register** button.

Here are sample steps how to download the ID file of the newly generated person from the Person document in the Domino Directory:

- Open the Domino Administrator.
- Open the **People & Groups** page.
- In the left navigation panel open the **People** node.
- Select the IDIPWSyncSigner person.
- Click **Edit Person** (this will open the Person document).
- At the bottom left corner of the Basics page, there is an attached file named "UserID". Open a context menu for that attachment and select **Save**.
- Click **Cancel** to close the Person document without changes.

The signer must have Manager access to the `pubnames.ntf` and `admin4.ntf` templates. Here is how to configure it for `admin4.ntf` (do the same for `pubnames.ntf`):

- Open the Domino Administrator.
- Open the **Files** page.
- In the **Show me** combo box select **Templates only**.
- In the list of templates, select `admin4.ntf`, open a context menu and select **Access Control -> Manage**.
- Click **Add...**
- Choose the IDIPWSyncSigner person.
- In the **User type** combo box, select **Person**.
- In the **Access** combo box select **Manager**s.
- Click **OK** to close the Access List window.

Finally you have to allow the signer to "sign or run unrestricted methods and operations":

- Open the Domino Administrator.

- Open the **Configuration** page.
- Select **Server -> All Server Documents**.
- Select the document of the Server (if you have multiple Domino servers, you have to apply the whole procedure to each one of them).
- Click **Edit Server**.
- Open the **Security** page.
- In the **Programmability Restrictions** section, add the signer person to the "Sign or run unrestricted methods and operations" field.
- Click **Save & Close** to save the changes to the Server document.

Deployment on Domino servers

The Domino HTTP Password Synchronizer can be deployed in the following modes:

- Both administrative password resets and normal user password changes are intercepted.
- Only normal user password changes are intercepted.
- Only administrative password resets are intercepted.

For deployment and configuration of the Domino HTTP Password Synchronizer see the following sections.

Deployment on a single Domino Server

To install the Domino HTTP Password Synchronizer on Domino, run the installer on the machine where the Domino Server is installed. The installer places all required files in the appropriate directory structures.

The file paths of Domino Server directories are as follows:

- The Domino Server Program Folder is referred to as *domino_program_directory* (for example, `C:\Program Files\IBM\Lotus\Domino` on Windows, `/opt/ibm/lotus` on Linux and UNIX-based platforms).
- The Domino Server Data Folder is referred to as *domino_data_directory* (for example, `C:\Program Files\IBM\Lotus\Domino\Data` on Windows, `/local/notesdata` on Linux and UNIX-based platforms).
- The Domino Server JVM Folder is referred to as *domino_jvm_directory* (for example, `C:\Program Files\IBM\Lotus\Domino\jvm` on Windows, `/opt/ibm/lotus/notes/80000/linux/jvm` on Linux and UNIX-based platforms).

Setup of the Domino plug-in

Do the following to set up the Domino plug-in:

1. Restart the Domino Server.
2. Sign `pwsync_install_r8.nsf` and `idipwsync.nsf` with Server ID:
   a. Start Lotus Domino Administrator.
   b. Select **Files**.
   c. Right-click on the **pwsync_install_r8** database and select **Sign**.
   d. In **Sign Database**, under **Which ID do you want to use?**, select **Active Server's ID**.
   e. Right-click on the **idipwsync** database and select **Sign**.
   f. In **Sign Database**, under **Which ID do you want to use?**, select **Active Server's ID**.
   g. Click **OK**.

3. Update the design of the `pubnames.ntf` template:
   a. Start Lotus Domino Designer.
   b. Open the following items:
      1) Open **pwsync_install_r8.nsf** database.
      2) Open `pubnames.ntf` template.
   c. Copy Agents:
      1) In `pwsync_install_r8.nsf`, select **Code/Agents**.
      2) Select both **IDIPWSyncClientAgent** and **IDIPWSyncWebAgent** (press the **Ctrl** key while clicking the two agents).
      3) Right-click on the selected agents and select **Copy**.
      4) In `pubnames.ntf`, select **Code/Agents**.
      5) Select **Edit -> Paste** to paste the two agents.

      If the Person form has not been modified with user-customized logic, the Person form from the Password Synchronizer is used.
   d. Rename the Person form in `pubnames.ntf`:
      1) In `pubnames.ntf` select **Forms**.
      2) Open the **Person** form.
      3) Select **Design -> Form Properties**.
      4) Edit the **Name** field. Change the name to **original_Person** (or other name of your choice, other than **Person**).

         Note: Make sure the default alias Person is also unset from that field.
      5) Save the form.
      6) Close the form.
   e. Copy the Person form:
      1) In `pwsync_install_r8.nsf` select **Forms**.
      2) Right-click on the **Person** form and select **Copy**.
      3) In `pubnames.ntf` select **Forms**.
      4) Select **Edit -> Paste** to paste the form.

      If the Person form has been modified with user-customized logic that needs to be kept, Password Synchronizer source code for the Person form must be copied manually.
   f. Copy the Person form source code:
      1) Copy **WebQuerySave** event code:
         a) In `pwsync_install_r8.nsf`, select **Forms**.
         b) Open the **Person** form.
         c) Select the **WebQuerySave** event.
         d) Copy the lines starting with `REM {start of IDI Password Synchronizer code};` and ending with `REM {end of IDI Password Synchronizer code};`
         e) In `pubnames.ntf` select **Forms**.
         f) Open the **Person** form.
         g) Select the **WebQuerySave** event.
         h) Paste the copied source code. Make sure the pasted code appears before any other code in this event.
         i) Save the form.
      2) Copy **QuerySave** event code:

a) In `pwsync_install_r8.nsf`, select **Forms**.

b) Open the **Person** form.

c) Select the **QuerySave** event.

d) Copy the lines starting with `'start of Password Synchronizer code` and ending with `'end of Password Synchronizer code`.

e) In `pubnames.ntf`, select **Forms**.

f) Open the **Person** form.

g) Select the **QuerySave** event.

h) Paste the copied source code. Make sure the pasted code appears just before the end of the **Querysave** procedure.

i) Save the form.

3) Copy **SyncPass** event code:

a) In `pwsync_install_r8.nsf`, select **Forms**.

b) Open the **Person** form.

c) Select the **SyncPass** event.

d) Copy all code for the **SyncPass** function.

e) In `pubnames.ntf`, select **Forms**.

f) Open the **Person** form.

g) Select the **QuerySave** event.

h) Paste the copied source code. Make sure the pasted code appears after all code in the event. A new event named **SyncPass** is created immediately, and the pasted code is transferred there.

i) Save the form.

If the **$PersonInheritableSchema** subform has not been modified with user-customized logic, the **$PersonInheritableSchema** from the Password Synchronizer is used.

g. Rename the **$PersonInheritableSchema** subform in `pubnames.ntf`:

1) In `pubnames.ntf`, select **Shared Elements/Subforms**.

2) Open the **$PersonInheritableSchema** subform.

3) Select **Design -> Subform Properties**.

4) Edit the **Name** field. Change the name to **original_$PersonInheritableSchema** (or other name of your choice other than **$PersonInheritableSchema**).

5) Save the form.

6) Close the form.

h. Copy the **$PersonInheritableSchema** subform:

1) In `pwsync_install_r8.nsf`, select **Shared Elements/Subforms**.

2) Right-click on the **$PersonInheritableSchema** form and select **Copy**.

3) In `pubnames.ntf`, select **Shared Elements/Subforms**.

4) Select **Edit ->Paste** to paste the subform. If the **$PersonInheriableSchema** subform has been modified with user-customized logic that needs to be kept, the Password Synchronizer source code must be copied manually.

If the **$PersonInheritableSchema** subform has been modified with user-customized logic that needs to be kept, Password Synchronizer source code is copied manually.

i. Copy the $PersonInheritableSchema subform code:

1) Copy **HTTPPassword** field code:
   a) In `pwsync_install_r8.nsf`, select **Shared Elements/Subforms**.
   b) Open the **$PersonInheritableSchema** subform.
   c) Select the **HTTPPassword** field (near the bottom of the form).
   d) Select the **Input Translation** event.
   e) Copy the lines starting with `REM {start of IDI Password Synchronizer code};` and ending with `REM {end of IDI Password Synchronizer code};`
   f) In `pubnames.ntf`, select **Shared Elements/Subforms**.
   g) Open the **$PersonInheritableSchema** form.
   h) Select the **HTTPPassword** field.
   i) Select the **Input Translation** event.
   j) Paste the copied source code. Make sure the pasted code appears before any other code in this event.
   k) Save the form.
2) Copy Enter Password button code:
   a) In `pwsync_install_r8.nsf`, select **Shared Elements/Subforms**.
   b) Open the **$PersonInheritableSchema** subform.
   c) Select the Enter Password button (near the bottom of the form).
   d) Select the Click event and make sure the Run field is set to "client".
   e) Copy the lines starting with `REM {start of IDI Password Synchronizer code};` and ending with `REM {end of IDI Password Synchronizer code};`
   f) In `pubnames.ntf`, select **Shared Code/Subforms**.
   g) Open the **$PersonInheritableSchema** form.
   h) Select the Enter Password button.
   i) Select the Click event. Again the Run field on the right hand side should be set to "client".
   j) Paste the copied source code. Make sure the pasted code appears after the piece of code where the received password (tmpPassword) gets verified and before the code that refreshes all the document fields (@Command([ViewRefreshFields]);)
   k) Save the form.
3) Copy **FullName** field code:
   a) In `pwsync_install_r8.nsf`, select **Shared Elements/Subforms**.
   b) Open the **$PersonInheritableSchema** subform.
   c) Select the **FullName** field (near the bottom of the form).
   d) Select the **Input Validation** event.
   e) Copy the lines starting with `REM {start of IDI Password Synchronizer code};` and ending with `REM {end of IDI Password Synchronizer code};`
   f) In `pubnames.ntf`, select **Shared Elements/Subforms**.
   g) Open the **$PersonInheritableSchema** form.
   h) Select the **FullName** field.
   i) Select the **Input Validation** event.
   j) Paste the copied source code before any other code in this event.
   k) Save the form.

4. Update the design of the `admin4.ntf` template:
   a. In Lotus Domino Designer open the `admin4.ntf` template database and `pwsync_install_r8.nsf` database.
   b. Copy the **IDIPWSyncAdminRequestAgent**:
      1) In `pwsync_install_r8.nsf`, select **Shared Code/Agents**.
      2) Select the **IDIPWSyncAdminRequestAgent**.
      3) Right-click on the selected agent and select **Copy**.
      4) In `admin4.ntf` select **Shared Code/Agents**.
      5) Select **Edit -> Paste** to paste the agent.
   c. Configure the **IDIPWSyncAdminRequestAgent**:
      1) Open the **IDIPWSyncAdminRequestAgent**.
      2) Select **Edit -> Properties**.
      3) Click **Edit settings** from the Runtime section of the Agent dialog box.
      4) In the **Run on** field select the name of the current Domino server.
      5) Click **OK**.
      6) Close the agent dialog box.
      7) Select **File -> Save** to save the new agent settings. You may get a warning message like "You do not have execution access privileges for agent 'IDIPWSyncAdminRequestAgent' on 'TDITest/IBM'; it will not run". The meaning of this message is that the Domino account that you use currently in the Domino Designer cannot "sign or run unrestricted methods and operations" on the Domino server. This is perfectly normal and it is why you will be instructed to sign the agents with a dedicated signer in the following steps.
5. Sign the agents with a signer that can "sign or run unrestricted methods and operations":
   a. Find the signer that is listed in the **Sign or run unrestricted methods and operations** field on the Security page of the Server document (to access the Server document, open the Domino Administrator, select Configuration and in the left navigation panel select Server/Current Server Document). If there are no existing accounts with this privilege, you may need to add a new one (there are sample steps to create a signer account in section “Creating a signer for the Password Synchronizer agents” on page 42. Beware that the privilege to "Sign or run unrestricted methods and operations" should be given only to the most trusted accounts. The signer that you choose must have Manager access to the `pubnames.ntf` and `admin4.ntf` templates, so that it can sign agents in them.
   b. Open the Domino Designer.
   c. Switch to the ID of the signer from step i. (**File -> Security -> Switch ID...**)
   d. Open the `pubnames.ntf` template.
   e. Select Code/Agents and open the list of all agents (at the top of the agents window you should see buttons **New Agent**, **Enable**, **Disable**, **Sign**)
   f. From the list of agents select **IDIPWSyncClientAgent**.
   g. Press the **Sign** button. (This will cause Domino Designer to sign the agent with the current ID.)
   h. From the list of agents select **IDIPWSyncWebAgent**.
   i. Press the **Sign** button.
   j. Open the `admin4.ntf` template.
   k. Select Code/Agents and open the list of all agents.

l. From the list of agents select the **IDIPWSyncAdminAgent**.

m. Press the **Sign** button.

n. Switch to the ID that you were using previously (**File -> Security -> Switch ID...**).

6. Refresh the design of the `names.nsf` database:

   a. In Lotus Domino Administrator, select **Files**.

   b. Select `names.nsf` database.

   c. Go to **File -> Application -> Refresh** Design.

   d. select the name of your server from the **With Design from Server** list.

   e. Click **OK**.

   f. Click **Yes** to continue.

7. Refresh the design of the `admin4.nsf` database:

   a. In Lotus Domino Administrator select **Files**.

   b. Select `admin4.nsf` database.

   c. Select **File -> Application -> Refresh Design**.

   d. Select the name of your server from the **With Design from Server** list.

   e. Click **OK**.

   f. Click **Yes** to proceed.

8. Setup secret key encryption infrastructure.

   Secret key encryption is used to protect passwords in the time slice in which they are temporarily stored in a database on the Domino Server.

   a. Generate a secret key:

      1) In Lotus Domino Administrator, select **File -> Security -> User Security**.

      2) Select **Notes Data/Documents** from the left navigation panel.

      3) Click **New Secret Key**.

      4) Enter **IdiPwSync** as secret key name and click **OK**.

      5) Click **Other Actions** and select **Export secret key**.

      6) Enter a password to protect the exported secret key.

         Note: This step is optional but highly recommended.

      7) Save the key in a file named `idipwsync.key`.

      8) Click **Close** in the **User Security** screen.

   b. Import the secret key in the Domino Server ID file:

      1) Stop the Domino Server.

      2) In Lotus Domino Administrator, select **File -> Security -> Switch ID**.

      3) Open the `server.id` file for the Domino Server. To do so you must use either a Lotus Domino Administrator installed on the Domino Server machine, or copy the `server.id` file to the machine where Lotus Domino Administrator is installed. The `server.id` file is usually placed in *`domino_data_directory`*.

      4) Select **File -> Security -> User Security**.

      5) Select **Notes Data/Documents** from the left navigation panel.

      6) Click **Other Actions** and select **Import secret key**.

      7) Open the `idipwsync.key` file.

8) If the file is password protected, enter the password that was created when you exported the secret key (see "Enter a password to protect the exported secret key," previous).

9) Click **Accept** to import the secret key.

10) Click **Close** in the **User Security** screen.

11) Select **File -> Security -> Switch ID** and switch back to the administrator ID file.

12) If you edited a copy of the `server.id` file, copy it over the original `server.id` file in *`domino_data_directory`* (you may want to backup the original `server.id` before overwriting it with the new one).

13) Start the Domino Server.

c. Import the secret key in the ID files of all Administrators or users that can edit Person documents and change http passwords. For each of these Administrators or users, do the following steps:

1) In Lotus Domino Administrator, select **File -> Security -> Switch ID**.

2) Open the ID file of the Administrator or user.

3) Select **File -> Security -> User Security**.

4) Select **Notes Data/Documents** from the left navigation panel.

5) Click **Other Actions** and select **Import secret key**.

6) Open the `idipwsync.key` file.

7) If the file is password protected, enter the password that was created when you exported the secret key (see the steps for generating a secret key in step 8a on page 48 above).

8) Click **Accept** to import the secret key.

9) Click **Close** in the **User Security** screen.

Note: An Administrator or user whose ID file does not contain the secret encryption key is not allowed to change the HTTP Password field of Person documents.

9. Setup port encryption (optional).

Port encryption encrypts the communication between Lotus Domino Administrator and the Domino Server, bringing an additional layer of security to the network communication.

Note: Port encryption is recommended but not required. Prior to being sent over the network, the password is encrypted with the secret key, regardless of whether port encryption is used or not.

Two options are available:

- Setup the Domino Server to encrypt communication ports. This is easier to set up (the Server settings only are configured) but it affects the communication with all clients, including regular users using Lotus Notes clients.
- Setup the Lotus Domino Administrator clients to encrypt communication ports. This requires configuration of each Lotus Domino Administrator client that is used, but does not affect other Notes clients if encryption is not necessary for them. Do this as follows:

a. Encrypt Domino Server communication ports:

1) In Lotus Domino Administrator select **Configuration**.

2) Select **Server/Server Ports** from the right-side panel.

        3) For each communication port in use, select the port in the **Communication ports** list and check the **Encrypt network data** option.
        4) Click **OK**.
        5) Restart the Domino Server for changes to take effect.
    b. Encrypt Lotus Domino Administrator communication ports:

    Do the following for each Lotus Domino Administrator client that is to be used for password changes:

        1) In Lotus Domino Administrator select **File -> Preferences ->-User Preference**.
        2) Select **Ports** from the left navigation panel.
        3) For each communication port in use, select the port in the **Communication ports** list and check the **Encrypt network data** option.
        4) Click **OK**.
        5) Restart Lotus Domino Administrator for changes to take effect.

10. Setup SSL for Domino HTTP Server.

    SSL is necessary to secure the communication between the Web browser and the Domino HTTP Server. If SSL is not set up, the password is transferred over the network in plain text.

    Consult the Lotus Domino Administrator help documentation for more information about setting up SSL ("Setting up SSL on a Domino server" is a recommended article).

11. Configure the Domino Server to automatically start and stop the Proxy Process:

    Open the file *domino_program_directory*/`notes.ini` and find the **ServerTasks** property. Add the following value at the end of the **ServerTasks** property:

    ```
    runjava com.ibm.di.plugin.domino.ProxyLoader
    ```

    The following is a sample **ServerTasks** property in `notes.ini`:

    ```
    ServerTasks=Update,Replica,Router,AMgr,AdminP,CalConn,Sched,HTTP,runjava com.ibm.di.plugin.pwsync.domino.ProxyLoader
    ```

12. Configure the Execution control list of Lotus Domino Administrator clients:

    Do the following for each Lotus Domino Administrator client that is to be used for password changes:

    a. In Lotus Domino Administrator select **File -> Security -> User Security**.
    b. Select **What Others Do/Using Workstation** in the left navigation panel.
    c. In the **When code is signed by** list, select the name of your Domino Server, for example **serverName/certifierName**. If the name of your Domino Server is missing, add it to this list.
    d. Under **Allow access to:**, check the **current database** option.
    e. Under **Allow ability to:**, check the **read other databases** and **Modify other databases** options.
    f. Click **OK**.

13. Configure Access Control:

    a. Create **IDIPWSync** group in the Domino Directory:

        1) In Lotus Domino Administrator, select **People & Groups**.
        2) In the left navigation panel, select **Domino Directories/***your_domain***'s Directory/Groups** where *your_domain* is the name of the Lotus Domino domain.

        3) Click **Add Group**.
        4) Type **IDIPWSync** in **Group name**.
        5) In the **Members** field add all Administrators or users that can change passwords by editing Person documents.
        6) In the **Members** field add the signer that you used to sign the agents of the Password Synchronizer.
    b. Update Access Control List of the `idipwsync.nsf` database:
        1) In Lotus Domino Administrator, select **Files**.
        2) Select the `idipwsync.nsf` database.
        3) Select **Database/Manage ACL** from the right-side panel.
        4) Click **Add** and select the **IDIPWSync** group.
        5) Set **Access** to **Editor**.
        6) Set the following options under Attributes:
            - Check the **Delete Documents option**. The options **Create Documents**, **Read Public Documents**, and **Write Public Documents** must be checked as well. This is done automatically when **Editor** access is selected.
            - Uncheck the options **Create private agents**, **Create personal folders/views**, **Create shared folders/views**, **Create LotusScript/Java agents**, **Replicate** or **copy documents**.
        7) Select **Default** from the **Access Control List**.
        8) Set **Access** to **No Access**.
        9) Click **OK**.

        Note: After the `idipwsync.nsf` database ACL is changed, it is no longer possible to change this ACL from the Domino Server. For security reasons, the most restrictive settings are used. If a new change of the ACL is necessary, the database must be opened locally and its ACL changed as needed.

14. Delete the `pwsync_install_r8.nsf` database. After the installation is complete, delete the `pwsync_install_r8.nsf` database from the Domino server:
    a. In Lotus Domino Administrator, select **Files**.
    b. Right-click on the `pwsync_install_r8` database and select **Delete Database**.
    c. Click **OK** in the **Confirm Database Delete** screen.

Notes:

1. The Domino HTTP Password Synchronizer ships with a template configuration file (`*TDI_install_dir*/pws_plugins/domino/pwsync.props`) that has all the required properties preset by default to enable out-of-the-box usage.
2. The default Password Store that is configured in the shipped `pwsync.props` file is the Log Password Store. This Password Store will log all the captured passwords in the proxy's log file. This password store should be used for diagnostic purposes only! Please refer to the individual Password Stores for more information on configuring them.

The table below explains the aforementioned steps in somewhat more detail:

Table 5. Explanation of customization steps

Step	Description
1	Make sure that the Domino Server has read the new files, which have been copied during the post-install phase.

Table 5. Explanation of customization steps (continued)

Step	Description
2	The external databases shipped with TDI need to be signed by the Domino server in order it to be able to vouch for their integrity.
3	By editing the `pubnames.ntf` template we change the behavior of the `names.nsf` database. A code is placed on several key places in order for the plain password to be intercepted. Once the password is captured it is passed to the responsible Java Agent (IDIPWSyncClientAgent or IDIPWSyncWebAgent).
4	By editing the admin4.ntf template we change the behavior of the `admin4.nsf` database. The copied Java Agent (IDIPWSyncAdminRequestAgent) is responsible for periodically processing the Administration Requests, posted by the various users when they change their passwords.
5	Agents execute with the rights of their signer. The agents of the Password Synchronizer need to perform restricted operations (network access, file system access), so they need to be signed by someone who can "sign or run unrestricted methods and operations".
6	Refreshing the design of the `names.nsf` applies the changed template to the existing database.
7	Refreshing the design of the `admin4.nsf` applies the changed template to the existing database.
8	The various Java Agents use the database `idipwsync.nsf` to store documents which need further processing. In order to protect the documents in this database they need to be encrypted. The secret key created in this step is used in the database encryption process.
9	Port encryption encrypts the communication between Lotus Domino Administrator and the Domino Server, bringing an additional layer of security to the network communication.
10	SSL is necessary to secure the communication between the Web browser and the Domino HTTP Server. If SSL is not set up, the password is transferred over the network in plain text.
11	The Java Proxy is executed in the JVM shipped with Domino. It is started as a Server Task when the Domino Server is starting.
12	Configure each Lotus Domino Administrator client to enable Administrative password changing.
13	The IDIPWSync group contains a list of the users which have the rights to change other users' passwords. Usually only the Administrators should present in this group. Regular user will still be able to change their passwords through iNotes even if they don't belong to this group. Members of this group are the only ones that can access the `idipwsync.nsf` database. The `idipwsync.nsf` database is used to transfer data between Lotus script and the Password Synchronizer agents. The signer of the Password Synchronizer agents must also be added in the IDIPWSync group, so that the agents will have access to `idipwsync.nsf` (agents execute with the rights of their signer).
14	The `pwsync_install_r8.nsf` database is only used for distributing the required template objects. Once the Domino HTTP Plugin is properly setup the database is no longer required and can be safely deleted.

Deployment on a Domino Domain with multiple Domino Servers

In environments with multiple Domino Servers the Password Synchronizer is installed on each Domino Server which is a Primary Domino Directory Server in the Domino Domain.

The Password Synchronizer is not installed on Domino Servers which are Configuration Only Directory Servers.

The installation on the Primary Domino Directory Servers is performed as follows:

1. On the Primary Domino Directory Server that is the Administration Server for the Domino Directory, perform full installation of the password synchronizer as described in the previous section, "Deployment on a single Domino Server" on page 43.
2. For all the other Primary Domino Directory Servers, do the following steps:
   a. Run the Password Synchronizer installer to install the necessary files.
   b. Force replication with the first Primary Domino Directory Server where a full setup is performed:
      1) In Lotus Domino Administrator select **Server**.
      2) Select **Status**.
      3) In the right-hand panel, select **Server/Replicate** .
      4) In the **Which server do you want to replicate with?** field, enter the name of the first Primary Domino Directory Server where a full setup is performed.
      5) Click **Replicate**.
      6) Click **Done**.
   c. This step and all following setup instructions refer to the setup steps from the section "Deployment on a single Domino Server" on page 43:

      Skip steps 1, 2, 3 and 6. Domino Directory replication propagates the design updates from the first Primary Domino Directory Server where a full setup is performed.
   d. Skip steps 4 and 7. The **IDIPWSyncAdminRequestAgent** is triggered only on the Administration Server for the Domino Directory.
   e. Perform step 8, but skip step 8a on page 48. (creation of a secret key). Use the secret key created when setting up the Password Synchronizer on the first Primary Domino Directory Server.
   f. Perform steps 9, 10, and 11.
   g. Skip step 12.
   h. Perform step 13, skipping step 13a on page 50. (the creation of the **IDIPWSync** group).
   i. Perform step 14.

Partial Deployment of the Password Synchronizer

The Domino Password Synchronizer intercepts both administrative password resets (when an administrator edits a user's person document) and normal password changes (when a user changes his own password through the **Change Password** Web form from `domcfg.nsf` or through iNotes).

Each one of these two features (intercepting administrative password resets and user password changes) can be installed and used independent of the other.

1. Install Domino Password Synchronizer that only intercepts administrative password resets (performed through the Lotus Domino Administrator or through the Web browser interface):

   To install a password synchronizer that will only intercept administrative password resets, perform all the steps from the "Deployment on a single Domino Server" on page 43 section of this document, except steps 4 and 7.

   On step 5, skip opening `admin4.ntf` and signing the **IDIPWSyncAdminRequestAgent** agent.

   Steps 4 and 7 install the agent that intercepts normal user password changes.

   When installing the solution on a Domino Domain with multiple Domino Servers, follow the instructions from the "Deployment on a Domino Domain with multiple Domino Servers" on page 53 section of this document, but do not perform steps 4 and 7 when installing the synchronizer on the Administration Server.
2. Install Domino Password Synchronizer that only intercepts normal user password changes (performed through the **Change Password** Web form from `domcfg.nsf` or through iNotes):

   To install a password synchronizer that intercepts normal user password changes only, perform the following steps from the "Deployment on a single Domino Server" on page 43 section of this document: 1, 2, 4, 5, 7, 10, 11 and 14. On step 5, skip opening `pubnames.ntf` and signing the **IDIPWSyncClientAgent** and **IDIPWSyncWebAgent** agents. The steps skipped (3, 6, 8, 9, 12 and 13) are not performed because they are only necessary for interception of administrative password resets.

   When installing the solution on a Domino Domain with multiple Domino Servers, it is only necessary to perform the previous subset of installation steps on the Primary Domino Directory Server which is the Administration Server for the Domino Directory. No installation on the other Domino Servers in the Domino Domain is necessary.

Deployment alternative that does not involve a dedicated agent signer (default pre-v7.1)

To minimize the scope of required privileges, in Tivoli Directory Integrator v7.1 the deployment procedure was modified to involve a dedicated signer account to sign the agents of the Password Synchronizer. The pre-v7.1 deployment procedure did not have such signer account, but instead required the IDIPWSync group to be given the privilege to "sign or run unrestricted methods and operations". The pre-v7.1 deployment procedure is still supported (although not recommended), so existing customers are encouraged but not forced to migrate.

To use the old deployment procedure apply the following modifications to the steps from section "Deployment on a single Domino Server" on page 43:

1. Skip step 5. (signing the agents)
2. Skip step 13.a) vi) (adding the signer account to the IDIPWSync group)
3. After step 13.a) perform the following steps (in multi-server topology apply these steps to all servers, where you deploy the Password Synchronizer):
   a. Select the **Security** page.
   b. In the field **Run unrestricted methods and operations** add the IDIPWSync group.
   c. Click **Save & Close** to save the changes to the server document.

Using the Password Synchronizer

Overview

The Domino HTTP Password Synchronizer modifies the "names.nsf" database, adding custom Java agents and custom code in certain hooks.

The code in these hooks is executed by Domino when a Person document is saved in "names.nsf". This code retrieves the http password before it is hashed and sends the password value to the Password Synchronizer Proxy Process using custom Java code.

The Domino HTTP Password Synchronizer modifies the administration requests database "admin4.nsf" by adding a custom Java agent. The agent is configured as a scheduled agent that is triggered after documents are created or modified in the administration requests database "admin4.nsf". The agent is not triggered immediately after a document is created/modified in "admin4.nsf", but after a 5min – 30min interval, depending on a decision made by the Agent Manager process in Domino. When triggered, the agent searches the admin request for successfully processed "Change HTTP password in Domino Directory" administration requests, retrieves the new passwords from these requests and sends the password data to the Password Synchronizer Proxy Process.

The Proxy Process invokes a Password Store component to encrypt and store the password data so that it can be retrieved by IBM Tivoli Directory Integrator.

Only certain password change mechanisms are intercepted

When using the Domino HTTP Password Synchronizer, be aware that only certain password change mechanisms are intercepted. These are the mechanisms listed previous:

- Editing the Person document through the Lotus Domino Administrator
- Editing the Person document through the Web browser
- Using the Change Password Web form from `domcfg.nsf`
- Using iNotes

Note: Password changes performed through any other interfaces are not intercepted. For example, if passwords are changed through LDAP, or a Notes-Internet password synchronization is enabled, the Domino HTTP Password Synchronizer is not triggered and these password changes are not synchronized.

Solution workflow

A Proxy Process is started by Domino when the Domino Server is started. This Proxy Process is configured to instantiate a Password Store (LDAP, JMS, and so forth). The Proxy Process accepts TCP/IP connections, receives user id and password data and invokes the Password Store to store this data.

Editing the Person document through the Lotus Domino Administrator

Custom code is placed in the "Person" form in the "names.nsf" database.

When the Person document is saved, this code is executed on the client (Lotus Domino Administrator). If the http password is changed, the following sequence of actions is performed:

1. The password is retrieved before it is hashed.
2. A new document is created and the password is stored in this document. The document is saved in a database on the server.
3. An agent is started on the server and passed the id of the newly created document. The agent reads the password data from the document, then deletes the password data from the document and sends the password data to the Proxy Process, which in turn sends it to the Password Store.
4. If the Password Store returns that the password has not been successfully stored, all changes made to the Person document will be rejected, including the change of the http password field.

Editing the Person document through the Domino web browser interface

Custom code is placed in the "Person" form in the "names.nsf" database. This code is executed on the Domino Server:

1. When a save of the Person document is requested and the http password value is changed, custom Lotus Formula code intercepts the plain text password and stores it in a custom hidden field in the document.
2. Just before the document is actually saved, Lotus Formula code starts an agent.
3. The agent reads the password value from the hidden field, deletes the value of this field and sends the password to the Proxy Process, which in turn sends it to the Password Store.
4. If the Password Store returns that the password has not been successfully stored, all changes made to the Person document will be rejected, including the change of the http password field.

Note: In this scenario the plain text password value is sent from the browser to the Domino Web Server when the web form is submitted. In order to protect the password on the wire, SSL is enabled on the Domino Web Server and users will use the HTTPS protocol from the browser.

Password change through the Password Change web form from domcfg.nsf or through iNotes

Change of the HTTP password through the Password Change web form or through iNotes results in a "Change HTTP password in Domino Directory" admin request posted in the "admin4.nsf" database. The Admin Process processes this requests and changes the password in the user’s Person document.

The password synchronizer adds a custom Java agent in the "admin4.nsf" database. After an administration request document or a reply to an administration request document is added to the "admin4.nsf" database, the Java agent is scheduled to start by the Agent Manager. The Java agent is not started immediately but after some configurable interval chosen by the Agent Manager (usually between 5 and 30 minutes after a document is posted). When the agent is run it performs the following actions:

1. Retrieves for processing all admin requests, which:

   are of type "Change HTTP password in Domino Directory"; and

   have already been processed successfully by the Domino Admin Process, that is, have a reply document attached that confirms that Domino has successfully changed the password (if a password change request has not been processed yet, or has not been processed successfully, then the password change has not been applied and thus there is no need for the password synchronizer to report it); and have not already been processed successfully by the agent on a previous run of the agent.

2. For each successfully processed password change admin request:

   The user identifier and the new password are retrieved and sent to the Proxy Process, which in turn sends them to the Password Store.

   If the Password Store returns that the password has been successfully stored, the admin request is marked as processed, so the agent would not process it again on the next run. If the password has not been successfully stored, the document is not marked as processed, so the agent will process it again on the next run.

Note: In this scenario the plain text password value is sent from the browser to the Domino Web Server when the web form is submitted. In order to protect the password in transit, SSL is enabled on the Domino Web Server and users will use the HTTPS protocol from the browser.

Secure password transfer

Secure communication is achieved by enabling SSL for the Web-based mechanisms for password change (editing Person documents through the browser, using the Change Password Web form and using iNotes).

When editing Person documents through the Lotus Domino Administrator client, communication is secured by enabling port encryption in Domino.

For instructions on how to configure port encryption for Domino, see “Setup of the Domino plug-in” on page 43 (specifically step 8).

Migration

No migration steps for pre-7.0 installations are available! A clean install of the plug-in is advised.

Migrate from v7.0 to v7.1

Starting from Tivoli Directory Integrator v7.1, the agents of the Password Synchronizer are signed by a dedicated signer who has the privilege to "sign or run unrestricted methods and operations". The IDIPWSync group is no longer required to have this privilege. The old deployment procedure (see section “Deployment alternative that does not involve a dedicated agent signer (default pre-v7.1)” on page 54) is still supported (but not recommended), so you can skip this migration step:

1. Sign the agents of the Password Synchronizer (see step 5 from section “Deployment on a single Domino Server” on page 43).
2. Refresh the designs of `names.nsf` and `admin4.nsf` (see steps 6 and 7 from section “Deployment on a single Domino Server” on page 43).
3. Add the signer of the agents to the IDIPWSync group (see step 13 from section “Deployment on a single Domino Server” on page 43).
4. Take away the privilege to "sign or run unrestricted methods and operations" from the IDIPWSync group:
   a. Open the Domino Administrator.
   b. Open the **Configuration** page.
   c. Select **Server -> All Server Documents**.
   d. Select the document of the Server (if you have multiple Domino servers, you have to apply the whole procedure to each one of them).
   e. Click **Edit Server**.

f. Open the **Security** page.

g. In the **Programmability Restrictions** section, remove the IDIPWSync group from the **Sign or run unrestricted methods and operations** field.

h. Click **Save & Close** to save the changes to the Server document.

Chapter 8. Password Synchronizer for UNIX and Linux

This chapter describes the configuration and operation of the IBM Tivoli Directory Integrator PAM Password Synchronizer. This chapter contains the following sections:

- "Overview"
- "Supported Platforms"
- "Deployment" on page 60
- "Configuration" on page 61
- "Reference Material" on page 62

Overview

The Pluggable Authentication Modules (PAM) architecture on UNIX systems, provides an extendable design to enable customized behavior with respect to user authentication. The PAM Password Synchronizer Plug-in leverages the UNIX PAM architecture to enable password change notifications to be propagated to the Tivoli Directory Integrator Plug-in Password Store.

There are numerous online documentation sites describing all aspects of PAM. The AIX online manual has a good overview description at the following link:

http://publib16.boulder.ibm.com/pseries/en_US/aixbman/security/pam_overview.htm#plugauthmod.

The primary purpose of the PAM Password Synchronizer Plug-in is to intercept password change events that originate from UNIX based tools and PAM enabled applications, like the "passwd" command.

Supported Platforms

The PAM Password Synchronizer is available for the following platforms:

- Solaris 9 SPARC (32–bit and 64–bit)
- Solaris 10 SPARC (64–bit)
- AIX 5L 5.3 (32–bit and 64–bit)
- AIX 6.1 (32–bit and 64–bit)
- RHEL ES/AS 4.0 (x86/x86–64)
- RHEL ES/AS 5.0 (x86/x86–64)
- SLES 9 (x86/x86–64)
- SLES 10 (x86/x86–64)
- RedFlag Data Center 5.0 SP1/Asianix 2.0 SP1

Notes:

1. On 64-bit x86 Linux, problems with the bundled JRE may be experienced if the Plug-in install is attempted before **prelink** has been run by **cron** for the first time. If this is the case, the Plug-in installation will fail with a message stating no JVM was found. Running the `/etc/cron.daily/prelink` script should resolve the issue and allow the Plug-in installation to proceed.

2. RHEL 5.0 has SELinux enabled by default. SELinux helps to keep the host secure from some sorts of malicious attacks. However default settings may prevent some of the plug-in libraries from loading. To fix this, SELinux must be told of the context under which these libraries run. To do this run the following command:

   ```
   find TDI_install_dir/jvm/jre/bin TDI_install_dir/pwd_plugins/PAM -name '*.so' -exec chcon -t textrel_shlib_t {} \;
   ```

Deployment

The Password Synchronizer is installed using the IBM Tivoli Directory Integrator installer wizard. After the installation is finished follow the steps in this section, which describes the deployment steps required for the PAM Password Synchronizer.

Registering the Password Synchronizer for UNIX and Linux plug-in within PAM

To register the plug-in, edit the PAM configuration file. The table below shows the standard location of both PAM configuration files on various platforms. Your individual PAM configuration may cause the PAM password module configuration to be a different file. You should check with your system administrator if either these files do not exist, or if the added Password Synchronization module is not being invoked.

Note: Older versions of PAM on UNIX used the configuration file /etc/pam.conf. This file is now deprecated and all PAM configuration files should now be located in /etc/pam.d for modules that rely on PAM. The PAM configuration file for the password change module should be located in this directory.

The primary component of external system configuration is the PAM configuration file. Since the purpose of the plug-in is to intercept password events, a line similar to the following should be added to the PAM configuration file. If the PAM module is being stacked with other PAM modules, then the Tivoli module should usually be the last in the stack. That way, the module can be sure that previous "required" modules have returned a success status before PAM called the Tivoli module.

Table 6.

Operating System	PAM Configuration File	PAM plug-in registration line
AIX 5.3 or greater	/etc/pam.conf	passwd password required *TDI_Plugin_Root*/pwd_plugins/pam/libpamtivoli.so use_first_pass *TDI_Plugin_Root*/pwd_plugins/pam/pwsync.props
Solaris 9,10	/etc/pam.conf or /etc/pam.d/system-auth	other password required *TDI_Plugin_Root*/pwd_plugins/pam/libpamtivoli.so use_first_pass *TDI_Plugin_Root*/pwd_plugins/pam/pwsync.props
Linux	/etc/pam.conf or /etc/pam.d/system-auth (RHEL 4) /etc/pam.conf or /etc/pam.d/password (SLES 9) /etc/pam.conf or /etc/pam.d/common-password (SLES 10)	password required *TDI_Plugin_Root*/pwd_plugins/pam/libpamtivoli.so use_first_pass *TDI_Plugin_Root*/pwd_plugins/pam/pwsync.props

Note: If the system is 64 bit (and the applications that rely on PAM, such as "passwd", are also 64 bit) you should use "libpamtivoli_64" instead of "libpamtivoli".

Note: The above table list **system-auth** as the PAM configuration file in the /etc/pam.d directory. In actual fact, the configuration file /etc/pam.d/passwd is the main configuration file for password setting and changing. On most operating systems, the standard PAM install sets up /etc/pam.d/passwd to use /etc/pam.d/system-auth for defining the actual PAM modules use for password setting and changing. For example on RHEL 4, the delegation in the /etc/pam.d/passwd file might look as follows.

```
password  required  pam_stack.so  service=system-auth
```

If your PAM /etc/pam.d/passwd configuration file has delegated to **system-auth**, then you must add the configuration entry into /etc/pam.d/system-auth.

There are exceptions to the placement of the Tivoli module last in the stack:

- If there are modules above the Tivoli module in the stack, and they are marked as *sufficient*, then they must be changed to *required* to ensure that the Tivoli module is called. For example on RHEL 4 Linux you may find that the pam_unix module is marked as *sufficient* (standard installation). This means that if the result of the pam_unix module is successful, then no proceeding password modules will be invoked. To ensure that the Tivoli module is called, the pam_unix module must be changed to *required* and it must appear before the Tivoli module in the stack.
- If you have modules for error processing only, such as pam_deny, then they should follow the Tivoli module, and the Tivoli module should be then marked as *sufficient*.

The PAM pluggable architecture allows the modules to be stacked. This means that custom solution can be created that allows several PAM Password Synchronizers to be installed on the same machine. Note each PAM plug-in would require a separate Java Proxy process (each Java Proxy should listen on a separate port). It is also recommended to use different pwsync.props files (at least they should not be in the same folder, because that folder is where the authentication is taking place).

Configuration

This section describes the configuration steps required for the PAM Password Synchronizer.

The PAM plug-in has a template configuration file installed at *TDI_Install_dir*/pwd_plugins/pam/pwsync.props. When the PAM plug-in is initialized, it will expect that the configuration file is set as the last parameter of the registration line of the plug-in. The plug-in then reads the file. Some of the parameters in that configuration file are shared between the plug-in and the Java Proxy. It recognizes some of the properties described in Chapter 3, “Password plug-ins common configuration and utilities,” on page 13.

The properties syncBase and logFile are irrelevant to the plug-in, thus they are ignored. The reason for the ignored syncBase property is that the PAM not always provides a dn-like naming of arrived users. The reason for the ignored logFile property is that the PAM plug-in always logs using the native UNIX syslog daemon.

Select the Password Store of your choosing by setting the correct class name in the `syncClass` parameter.

Reference Material

For information on enabling PAM on AIX 5.3 see Appendix A, “AIX 5.3 PAM Configuration,” on page 87.

Chapter 9. LDAP Password Store

The LDAP Password Store provides the function necessary to store intercepted user passwords in an LDAP directory server.

This chapter contains the following sections:
- "Supported Directories"
- "Installing LDAP Password Store"
- "Using the Password Store" on page 69

Supported Directories

The LDAP Password Store is available on the following directories:
- IBM Tivoli Directory Server
- Microsoft Active Directory
- Sun Directory Server

Installing LDAP Password Store

This section describes the LDAP Password Store installation process, including prerequisites.

Overview

IBM Directory Integrator LDAP Password Store provides the function necessary to store the intercepted user passwords in an LDAP directory server (repository or datasource). Supported directories include IBM Directory Server, Microsoft Active Directory and Sun Directory Server.

The LDAP Password Store component of this package was created to support a growing number of IBM Directory Integrator plug-ins which intercept password changes for various products or platforms.

The following password synchronization plug-ins are available to intercept a user's password change request:

IBM Directory Integrator Password Synchronizer for Windows
Intercepts the Windows login password change.

IBM Directory Server Password Synchronizer for Windows, UNIX and Linux
Intercepts an IBM Directory Server password change.

Sun Directory Server Password Synchronizer for Windows, UNIX and Linux
Intercepts the Sun Directory Server password change.

Domino Password Synchronizer for Windows, UNIX and Linux
Intercepts changes of the HTTP password for Lotus Notes users.

IBM Directory Integrator Password Synchronizer for UNIX and Linux
Intercepts changes of UNIX and Linux user passwords.

These plug-ins all utilize the LDAP Password Store function which facilitates the secure propagation of the change to another LDAP server where it can later be manipulated by an IBM Directory Integrator AssemblyLine.

The ability to tailor the LDAP Password Store is accomplished using properties files which enable the specification of keystore files, certificates and credentials for SSL connections and the asymmetric encryption of password data. The property files also accommodate control of trace logging, and limited control of attributes used for storing captured passwords.

Prerequisites

The LDAP Password Store requires as a minimum JRE 1.5; Tivoli Directory Integrator 7.1 bundles a Java 6 JRE.

Installing LDAP Password Store

Do the following to set up and install the LDAP Password Store:

Set up the LDAP server

The following instructions describe how to set up a sample environment using IBM Directory Server. This involves identifying a container where the object class containing the user ID and password is found or created.

Do the following to set up a sample environment using IBM Directory Server:

1. Define the suffix.
   a. Start **Directory Configuration**. Select **Start -> Programs -> IBM Directory Server x.x -> Directory Configuration**.
   b. Select **Manage suffixes** from the pane in the left.
   c. In the **Suffix DN** field add the suffix under which you store the password information (for example, `o=ibm,c=us`).
   d. Click **Add**.
   e. The new suffix is shown in the **Current suffix DNs** list. Click **OK**.
   f. Close the **Directory Configuration** tool.
2. Add the suffix data.
   a. Restart the IBM Directory Server.
   b. Using **IBM Directory Server Web Administration Tool**, select **Directory management>Manage entries**.
   c. Click **Add**.
   d. Select **organization** from the structural object class list.
   e. Click **Next**.
   f. In the **Select auxiliary object classes** screen, click **Next**.
   g. In the **Enter the attributes** screen, clear the value of the Parent DN field.
   h. Enter the suffix name into the **Relative DN** field (for example, **o=ibm,c=us**).
   i. Enter the organization name into the **o** field (**ibm** in the previous example).
   j. Click **Finish**.
3. Add the domain object.
   a. Still using the **IBM Directory Server Web Administration Tool**, select **Directory management -> Manage entries**.
   b. Select the suffix previously created in the previous step (**o=ibm,c=us**) by selecting the corresponding radio-button.
   c. Click **Add...**
   d. Select **domain** from the **structural object class** list.
   e. Click **Next**.
   f. In the **Select auxiliary object classes** screen, click **Next**.

g. Enter the domain name in the **Relative DN** field (for example, **dc=mydomain**).

h. Enter the domain name in the **dc** field (**mydomain** in the previous example).

i. Click **Finish**.

Note: The domain and suffix entered must also be included in the pwsync.props file along with the other information (see "Configuring the LDAP Password Store" on page 66 for more details).

4. Define the **ibm-diPerson** object. From a machine with IBM Directory Server Client, issue the following command from the *install_directory* (as one line):

```
ldapmodify -c -h LDAP Hostname -D admin DN -w admin PW
       -f ibm-diPerson_oc.ldif
```

Note: You may see the following messages:

```
attribute type '1.3.18.0.2.4.155' already exists, add operation
failed.
```

or

`attribute type '0.9.2342.19200300.100.1.1'  already exists, add operation failed`. You can ignore these messages, they indicate that these **secretKey** and **uid** attributes are already defined in your schema.

Modifying the schema of zLDAP

Note: When configuring the LDAP server on z/OS, the LDAP server must be configured with a TDBM back end (this enables loading of the required LDIF file). Detailed instructions for setup and configuration of the IBM LDAP server on z/OS with a TDBM back end are beyond the scope of this guide. For further information on this issue, please see the document *z/OS Integrated Security Services LDAP Server Administration and Use* in the IBM z/OS online product library.

Modify the schema of zLDAP as follows

1. Definition of a suffix involves generating a new LDAP config and server JCL Jobs. This must be performed jointly by the LDAP administrator and the SYS Programmers.
2. While the suffix data does not need to be added, the base schema does need to be added to the defined suffix. The two base schema LDIF files in the /usr/lpp/ldap/etc directory, schema.IBM.ldif and schema.user.ldif, must be customized with the suffix from step 1 above, and then loaded.
3. If required, a domain can be defined by creating and loading a LDIF file that defines the domain.
4. Before the ibm-diPerson_z.ldif file can be loaded into the LDAP server, it must be customized to include the suffix created in step 1. This involves adding the suffix to the end DN's. For example, if your suffix was "o=ibm,c=us" then the DN lines would change from this:

   ```
   dn:cn=schema
   ```

   To this:

   ```
   dn:cn=schema,o=ibm,c=us
   ```

Modifying the schema of Sun Directory Server and Active Directory

1. Modify the LDAP schema of Sun Directory Server. Issue the following command (as one line):

   ```
   ldapmodify -c -h LDAP Hostname -D admin DN -w admin PW
          -f ibm-diPersonForSunDS.ldif
   ```

2. Modify the LDAP schema of Active Directory.

   a. Enable Active Directory schema modification by editing the Windows registry key:

      ```
      HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\NTDS\Parameters
      ```

      Add a **REG_DWORD** value named **Schema Update Allowed** with a value of **1** (or any value greater than **0**).

   b. Issue the following command to update the LDAP schema:

      ```
      ldifde -i -f ibm-diPersonSchemaForAD.ldif
      ```

   c. Open the Microsoft Management Console.

   d. Create a new Organizational Unit. This is where the changed passwords are going to be stored.

   e. Get the Distinguished Name of the Organizational Unit using one of the following tools: "ldifde.exe", "csvde.exe", or "dsquery.exe". This will be needed when configuring the suffix of the LDAP Password Store in the `pwsync.props` file.

Configuring the LDAP Password Store

This section describes how to configure the LDAP Password Store.

Properties pertaining to the LDAP Password Store are set in the plug-ins general configuration file: `pwsync.props`. By default there is one file per each plug-in, for example, *`TDI_Install_dir`*`/pwd_plugins/tds/pwsync.props (for the IBM Directory Server Password plug-in). The LDAP Password Store is therefore configured in the pwsync.props file of the Password intercept plug-in you are using on that platform..`

Note: In the general configuration file, you must encrypt each password property manually. This can be done using the encryptPasswd utility. Be aware that this utility uses a symmetric algorithm for encryption of the passwords. Make sure that the `pwsync.props` file is readable only by trusted system users.

The encyptPasswd utility expects that the password is passed as a parameter. The encrypted password is printed on the standard output.

For a complete list of the configuration parameters and their explanation, see Chapter 3, “Password plug-ins common configuration and utilities,” on page 13.

The class for this password store is:
`com.ibm.di.plugin.pwstore.ldap.LDAPPasswordStore`.

An example of a completed properties file for an SSL connection and password encryption looks like the following:

```
#IBM Directory Integrator LDAP Password Store Settings with Encoded Passwords
#Tue Jul 30 08:21:20 EDT 2002
ldap.hostname=gbdthst1
ldap.port=636
ldap.waitForStore=true
ldap.admindn=cn=root
```

```
ldap.password=0c0bf0e3146b
ldap.ssl=true
ldap.suffix=dc=carnd11,o=ibm,c=us
encrypt=true
encryptKeyStoreFilePath=c:\sync\cryptokeys.jks
encryptKeyStoreFilePassword=0c0bf0e3146b
encryptKeyStoreCertificate=cryptoCertName
encryptKeyPassword=0c0bf0e3146b
```

Notes:

1. To disable SSL, select a non-SSL port (for example, 389) and set ssl=false.
2. To disable asymmetric password encryption, set encrypt=false. When encrypt=false, any value in encryptKeyStoreFilePath , encryptKeyStoreFilePassword, encryptKeyStoreCertificate and encryptKeyPassword is ignored.
3. The suffix keyword is used to identify the container where objects containing the user ID and new password value are found.
4. There are some additional optional keywords that can be used to override the default object class and attribute definitions provided. The following are the names of the properties that can be added in `pwsync.props` and their associated default values:

   ldap.schemaPersonObjectName
   ibm-diPerson

   ldap.schemaUseridAttributeName
   ibm-diUserId

   ldap.schemaPasswordAttributeName
   ibm-diPassword
5. Another optional attribute, **ldap.delayMillis**, is used when the ldap.waitForStore property is set to false. When ldap.waitForStore=false, ldap.delayMillis specifies the number of milliseconds of delay prior to performing the store. If the IBM Directory Integrator Password Synchronizer for Windows is configured to use the LDAP Password Store and the LDAP Password Store is configured to store into Active Directory on the same machine where the Password Synchronizer is installed, a deadlock can occur. To avoid deadlock, use this asynchronous mode of operation. In the asynchronous mode (ldap.waitForStore=false), the password catcher code which communicates with the Windows system returns control to Windows. After a short delay, the password store code which is running a separate thread attempts the store of the password update into Active Directory. If ldap.waitForStore=false and no value is specified for ldap.delayMillis, then a default of ldap.delayMillis=2000 is used. In this configuration, any password store failures are reported using the log file specified in the logFilePath property.

About encrypting passwords

Encryption of password values is supported by both the LDAP Password Store and the JMS Password Store.

By default encryption is disabled. To turn it on, set the `encrypt` property to true.

When encryption is used, *encryptKeyStoreFilePath*, *encryptKeyStoreFilePassword*, *encryptKeyStoreCertifcate* and values must also be set. Additionally the *encryptKeyPassword* property must be set if you are using the LDAP Password Store (see the remarks below for explanation of this requirement). The *encryptKeyPassword* property is irrelevant for the rest of the Password Stores. The password encryption and decryption functions use the RSA algorithm. Below is a reference of the configuration properties for the encryption functionality:

```
encryptKeyStoreFilePath=path to the key store file
encryptKeyStoreFilePassword=password of the key store file; encoded with the "encryptPasswd" tool
encryptKeyStoreCertifcate=the alias of the public key certificate in the key store
encryptKeyPassword=password of the private key; encoded with the "encryptPasswd" tool
```

See the remarks below for discussion of what keys should be present in the key store.

You can create and manage keystore files and public/private keys with the `keytool` and `Ikeyman` JRE utilities.

Information about keystores and keytools is available from the following sites:

- *IBM Tivoli Directory Integrator V7.1 Installation and Administrator Guide*, section Security and TDI -> Secure Socket Layer Support -> Keystore and truststore management
- http://www-128.ibm.com/developerworks/websphere/techjournal/0502_benantar/0502_benantar.html#sec2
- http://java.sun.com/j2se/1.5.0/docs/tooldocs/windows/keytool.html

The `java.security` file located in the *`install_directory`*`/jvm/jre/lib/security` directory has been set up to contain a reference to security provider `com.ibm.crypto.provider.IBMJCE` . The following is an example of how the relevant portion of the file might look:

```
                        :
                        :
                        :

# List of providers and their preference orders :
#
security.provider.1=sun.security.provider.Sun
security.provider.2=com.ibm.crypto.provider.IBMJCE

                        :
                        :
                        :
```

An example AssemblyLine which demonstrates the decryption of captured passwords is included in the Tivoli Directory Integrator installation. The AssemblyLine and a readme file are located in the *`TDI_install_dir`*`/examples/pwsync_decryption/` directory where *TDI_install_dir* is the install directory of the IBM Tivoli Directory Integrator.

Notes:

1. RSA is an asymmetric encryption algorithm – it uses a public key to encrypt and its associated private key to decrypt. Since you need just the public key to do encryption, it is strongly recommended to distribute only the public key in the key store file of the Password Store. This recommendation does not apply to the LDAP Password Store, because it also does decryption of already stored password values to determine which ones to delete (so it needs the private key too).
2. The key store files contain sensitive data and must be properly protected using file system permissions.

Using the Password Store

For each user whose password has been intercepted, the LDAP Password Store maintains an LDAP entry in the storage LDAP directory (the container where the storage entries are added and modified is specified by the suffix property of the LDAP Password Store).

The entry kept in the storage directory always contains the passwords currently used by the original user on the Target System. To achieve this, the LDAP Password Store updates the state of the entry in the storage directory whenever the LDAP Password Store receives notification for password update from the Password Synchronizer.

The LDAP Password Store receives the following data from the Password Synchronizer:

- User identifier (a string)
- Type of the password modification
- A list of password values

User Identifier:

The user identifier is used for the relative distinguished name of the entry stored in the LDAP directory. For example, if the user identifier is "john" and the suffix property value is "dc=somedc,o=ibm,c=us", then the distinguished name of the entry stored is "ibm-diUserId=john, dc=somedc,o=ibm,c=us".

Special attention is necessary when the LDAP Password Store is used with the IBM Tivoli Directory Server Password Synchronizer or with the Sun Directory Server Password Synchronizer.

The Password Synchronizer reports the LDAP distinguished name of the user for which the password has been changed. For example, "cn=john,o=somecompany,c=us". The LDAP Password Store takes the first element of the distinguished name ("john") to construct the distinguished name of the entry on the storage LDAP directory, for example, "ibm-diUserId=john, dc=somedc,o=ibm,c=us". Therefore the context information (department, company, country, and so forth) is lost. If there are two individuals on the Target System with equal names but in different departments, for example, "cn=Kyle Nguyen,ou=dept_1,o=ibm,c=us" and "cn=Kyle Nguyen,ou=dept_2,o=ibm,c=us", they are indistinguishable for the Password Store, and the Password Store acts as if they represent the same person.

Type of password modification and List of password values:

The type of password modification indicates whether the password values have been replaced, or new values have been added, or certain values have been deleted. Using this information and the list of passwords representing the change, the Password Store duplicates the change on the entry in the storage directory.

The type of password modification makes sense only when the password can have multiple values (IBM Tivoli Directory Server, Sun Directory Server). When the passwords on the Target System are single-valued (Windows), the password modification type is always **replace**.

When the password (with all its values) is deleted from the Target System, the entry in the storage directory is modified so that it does not have value for the LDAP attribute used to store the passwords.

Possible password retrieval from IBM Tivoli Directory Integrator:

Here is a possible mechanism for retrieving passwords stored in an LDAP Server by the LDAP Password Store:

A Changelog Connector is configured to listen for changes in the LDAP Directory used for storage. Whenever the Connector detects that an entry has been added or modified in the Password Store container, it starts an AssemblyLine, passing it identification of the modified entry. The AssemblyLine uses an LDAP Connector to read the modified entry, then decrypts the updated password values and propagates the values to systems that must be kept synchronized.

Chapter 10. JMS Password Store

This chapter contains the following sections:

Overview

JMS Password Store (formally known as the MQ Everyplace Password Store) provides the functionality necessary to store intercepted user passwords in a JMS Provider's Queue from where any JMS client for example, Tivoli Directory Integrator) could read them.

The JMS Password Store package consists of the Storage Component and the JMS Password Store Connector (for more information about the JMS Password Store Connector, refer to the *IBM Tivoli Directory Integrator V7.1 Reference Guide*.) The Storage Component is actually the Password Store invoked by the Password Synchronizer. The JMS Password Store Connector is a specialized Connector on the IBM Tivoli Directory Integrator side that can retrieve passwords stored by the configured JMS Provider.

The class for this password store is: `com.ibm.di.plugin.pwstore.jms.JMSPasswordStore`

IBM WebSphere MQe driver

The IBM WebSphere MQe driver is responsible for spawning the MQe Queue Manager and retrieving the required connection objects.

In order to use MQe as the JMS provider for the JMS Password Store component the jmsDriverClass property in the `pwsync.props` file must be set to `com.ibm.di.plugin.pwstore.jms.driver.IBMMQe`.

Note: An MQe Queue Manager needs to be created. This can be done using the MQe Configuration utility bundled with the Password Synchronizer. For more information about the MQe Configuration utility see section "MQe Queue Manager setup" on page 75.

IBM WebSphere MQ driver

The IBM WebSphere MQ driver is responsible for establishing the connection with the IBM WebSphere MQ JMS provider. In order to use MQ as the JMS provider for the JMS Password Store component the jmsDriverClass property in the `pwsync.props` file must be set to `com.ibm.di.plugin.pwstore.jms.driver.IBMMQ`.

IBM WebSphere MQ driver has the following parameters:

jms.broker
The MQ server address (IP address and TCP port number); an example value would be "192.168.113.54:1414"

jms.serverChannel
The name of the server channel configured for the MQ server instance

jms.qManager
The name of the Queue Manager defined for the MQ server instance

jms.sslCipher
The cipher suite name which corresponds to the cipher selected when configuring the MQ server channel; an example value is "SSL_RSA_WITH_RC4128_MD5"

jms.sslUseFlag
Specifies whether SSL will be used on the connection to the MQ Server instance; valid values are "true" and "false"

If SSL is going to be used with the MQ Driver then the properties listed in the table titled SSL Java Properties in Chapter 3, “Password plug-ins common configuration and utilities,” on page 13 must be used to establish a trustful relationship between the JMS Password Store (client) and Websphere MQ (server).

For specific configuration of the IBM WebSphere MQ server please refer to its documentation.

Microbroker Driver

For users with existing Microbroker (MB) installations, a Microbroker driver is provided, responsible for establishing the connection with the Microbroker provider. In order to use Microbroker as the JMS provider for the JMS Password Store component, the `jmsDriverClass` property in the `pwsync.props` file must be set to `com.ibm.di.plugin.pwstore.jms.driver.IBMMB`.

The MB driver has the following parameters:

jms.broker
the MB server address (IP address and TCP port number); an example value would be "9.126.6.120:1883"

jms.clientID
the client ID; it is required.

Note: In order to be able to use Microbroker as the JMS password store, some Microbroker jars are needed. A sample list of the required jars is available in section External System Configuration, Microbroker of the JMS Connector in *IBM Tivoli Directory Integrator V7.1 Reference Guide*.

JMS Script Driver

The JMS Password Store does not support user defined JMS Script Drivers as Tivoli Directory Integrator does. This is because no JavaScript™ engine is bundled with the Password Synchronizer.

Configuring the JMS Password Store

This section describes how to configure the JMS Password Store.

Properties pertaining to the JMS Password Store are set in the plug-ins general configuration file: `pwsync.props`. By default there is one file per each plug-in, for example, *`TDI_Install_dir`*`/pwd_plugins/tds/pwsync.props`

Note: In the general configuration file, you must encrypt each password property manually. This can be done using the encryptPasswd utility. Be aware that this utility uses a symmetric algorithm for encryption of the passwords. Make sure that the pwsync.props file is readable only by trusted system users.

The encyptPasswd utility expects that the password is passed as a parameter. The encrypted password is printed on the standard output.

For a complete list of the configuration parameters, their explanation and the encryptPasswd utility, see Chapter 3, "Password plug-ins common configuration and utilities," on page 13.

An extract of the JMS Password Store configuration section of the pwsync.props file follows:

```
# Passwords encryption properties:

### Specify true or false to correspondingly turn the
### encryption of passwords on or off.
encrypt=true

### The path of the JKS file that is used to encrypt
### passwords (only taken into account when encrypt
### is set to true).
encryptKeyStoreFilePath=

### The encrypted password of the JKS file (only
### taken into account when encrypt is set to true).
### This maps to the -storepass parameter for keytool
### -genkey
encryptKeyStoreFilePassword=

### The alias of the key from the JKS file.
encryptKeyStoreCertificate=

# PKCS7 Configuration:

### This indicates whether or not the option is turned on.
pkcs7=false

### The file path and the name of the JKS file.
pkcs7KeyStoreFilePath=

### The password for the JKS file.
pkcs7KeyStoreFilePassword=

### The alias of the MQePasswordStore's certificate.
pkcs7MqeStoreCertificateAlias=

### The alias of the MQePasswordStoreConnector's certificate.
pkcs7MqeConnectorCertificateAlias=

# The specific driver used for establishing connection with a broker.
# Possible values:
### com.ibm.di.plugin.pwstore.jms.driver.IBMMQe
### com.ibm.di.plugin.pwstore.jms.driver.IBMMQ
jmsDriverClass=com.ibm.di.plugin.pwstore.jms.driver.IBMMQe

# The ID of this client. This value is used when
# connecting to a broker. Most brokers do not allow
# clients to have the same ID.
jms.clientId=

# MQe Configuration:
```

```
### The path to the .ini file of the generated MQe QueueManager.
mqe.file.ini =

### The TCP/IP port that is used when the MQe Connector
### sends notifications to the Storage Component. Default
### value is 41002.
mqe.notify.port=41002

# Websphere MQ Configuration:

### JMS Server address (ip host and tcp port number).
jms.broker=<host>:<port>

### Login username for the password queue.
jms.username=

### Login password for password queue.
### Note: This field should be encoded. Use the following utility:
### encryptPasswd <yourpassword>.
jms.password=

### MQ Server Channel.
jms.serverChannel=

### Specifies MQ Queue Manager Name.
jms.qManager=

### If true, you must have a properly configured the JMS provider.
jms.sslUseFlag=false

### CipherSuite names supported by WebSphere MQ.
### Possible values:
### SSL_RSA_WITH_DES_CBC_SHA
### SSL_RSA_WITH_NULL_MD5
### SSL_RSA_WITH_NULL_SHA
### SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5
### SSL_RSA_WITH_RC4_128_MD5
### SSL_RSA_EXPORT_WITH_RC4_40_MD5
### SSL_RSA_WITH_RC4_128_SHA
### SSL_RSA_WITH_3DES_EDE_CBC_SHA
### SSL_RSA_EXPORT1024_WITH_RC4_56_SHA
### SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA
### SSL_RSA_WITH_AES_128_CBC_SHA
### SSL_RSA_WITH_AES_256_CBC_SHA
### SSL_RSA_FIPS_WITH_DES_CBC_SHA
### SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA
jms.sslCipher=SSL_RSA_WITH_RC4_128_MD5
```

In this section, the following parameters merit attention:

mqe.file.ini

Required if you are using the MQe driver, if not then this is ignored and the jms.broker property is used instead.

The path to the .ini file generated by the MQe Configuration Utility (usually C:\\Program Files\\IBM\\TDI\\V7.1\\pwd_plugins\\tds\\MQePWStore\\pwstore_client.ini).

mqe.notify.port

Required if you are using the MQe driver, if not then this is ignored.

The TCP/IP port that is used when the JMS Password Connector sends notifications to the MQe Driver on behalf of the JMS Password Store. Default value is 41002.

Note: For more information about the usage of this parameter, also see section "Force transfer of accumulated messages from the JMS Password Store with MQe" in *IBM Tivoli Directory Integrator V7.1 Reference Guide*.

MQe Queue Manager setup

Create and configure the MQe QueueManager.

The file `mqeconfig.jar` placed in *`TDI_Install_dir`*`/pwd_plugins/jars` contains a utility program (MQe Configuration Component) that automatically creates and configures the MQe QueueManager that is used by the Storage Component.

- Before running the MQe Configuration Component, open its properties file *`TDI_Install_dir`*`/pwd_plugins/etc/mqeconfig.props` and set values for the following properties:

  clientRootFolder
  The folder where you want to place the MQe QueueManager (for Windows, for example: `C:\\Program Files\\IBM\\TDI\\V7.1\\pwd_plugins\\tds\\MQePWStore`).

  Note: When specifying Windows filepaths in the property files, the backslash file separator (\ ) must be escaped with a second backslash (\\ ).

  serverIP
  This is the IP address of the machine where the IBM Directory Integrator and the JMS Password Store are deployed.

  communicationPort
  The TCP/IP port that is used for communication between the two MQe QueueManagers.

  clientRegistryType
  Optional. Required for authenticated MQe access deployments only. If used, value must be set to "PrivateRegistry". The Private Registry stores the certificates issued by the MQe Mini-Certificate server.

  clientRegistryPin
  Optional. Required for authenticated MQe access deployments only. If used, this value represents the "PIN" access code used by the Tivoli Directory Integrator JMS Password Store to access the PrivateRegistry. This value will be stored as plain text in the result MQe ".ini" file.

  clientKeyRingPassword
  Optional. Required for authenticated MQe access deployments only. This value is used when requesting a certificate from the MQe Mini-Certificate server It is the seed value for certificate generation. This value will be stored as plain text in the result MQe ".ini" file.

  certServerReqPin
  Optional. Required for authenticated MQe access deployments only. This value is used as a one time authentication PIN by this Queue Manager when requesting certificates from the MQe Mini-Certificate server. This value must match the "Request PIN" value from the Mini-Certificate server setup.

  certServerIPAndPort
  Optional. Required for authenticated MQe access deployments only. This value is used as the destination address for MQe Mini-Certificate server

requests. The format of the value is "FastNetwork:<host>:<port>", where host must be the machine name or TCP IP address where the MQe Mini-Certificate server is running, and port value must match the "Port" value from the Mini-Certificate server setup.

debug Specify **true** or **false** to correspondingly turn debug information on or off.

The following is a sample **mqeconfig.props** configuration file:

```
clientRootFolder=C:\\Program Files\\IBM\\TDI\\V7.1\\pwd_plugins\\tds\\MQePWStore
serverIP=127.0.0.1
#clientRegistryType=PrivateRegistry
#clientRegistryPin=<Private client registry access PIN>
#clientKeyRingPassword=<Seed value for certificate generation>


# Properties used for setting up MQe Queue Manager as server

serverRootFolder=C:\\Program Files\\IBM\\TDI\\V7.1\\MQePWStore
#serverRegistryType=PrivateRegistry
#serverRegistryPin=<Private client registry access PIN>
#serverKeyRingPassword=<Seed value for certificate generation>

#certServerReqPin=<One time certificate request PIN>
#certServerIPAndPort=FastNetwork:<Mini-Certificate server hostname or IP>:<port>
#certRenewalEntityName=<QueueManager name or QueueManager+Queue name>

communicationPort=41001

#disableQueueRegistry=
debug=true
```

Note: When specifying Windows filepaths in the property files, the backslash file separator (\ ) must be escaped with a second backslash (\\ ).

The **serverRootFolder** property is not used when configuring the Storage Component (it is used to configure the QueueManager at the MQe Connector) and its value is not taken into account here.

- To create and automatically configure MQe QueueManager for the Storage Component, open a command prompt in the *TDI_Install_dir*/pwd_plugins/bin folder and enter the following command (as one line):

  ```
  .\mqeconfig.bat ..\etc\mqeconfig.props create client
  ```

  The log of this command is displayed on the console. After successful completion, the message Client MQe configuration successfully completed displays. If the mqeconfig.props file contains the optional parameters for MQe authenticated access, this step will automatically request the necessary certificates from the MQe Mini-Certificate server.

  Tip: If attempting to perform an MQe certificate authenticated access deployment, it is important to remember that certificates may be requested once only per authenticate-able entity. If an exception message similar to the one below is reported during configuration, it may be necessary to re-enable certificate issue for that entity using the Mini-Certificate server GUI.

  ```
  [MQeConfig] [28/07/05 10:10:01]: Action failed:
  Code=351;com.ibm.mqe.MQeException: Registration exception =
  com.ibm.mqe.MQeException: certificate request failed[PWStoreClient 4]
   (code=8)[PWStoreClient 8] (code=351) [MQeConfig] [28/07/05 10:10:01]:
  Error: Server MQe configiration failed; exception:java.lang.Exception:
  Code=351;com.ibm.mqe.MQeException: Registration exception = com.ibm.mqe.
  MQeException: certificate request failed[PWStoreClient4] (code=8)
  [PWStoreClient 8] (code=351)
  ```

Note: If you need to change the configuration of the QueueManager, you have two options:

- Delete the QueueManager from the disk and create it again following the previous procedure, or
- Install an MQ Everyplace admin tool compatible with MQ Everyplace 2.0.2.5 QueueManagers (for example, MQe Explorer) and use it to change the QueueManager settings.

Websphere MQ setup

When IBM WebSphere MQ is used as JMS provider the following jar files have to be taken from the WebSphere MQ installation and included in the class path of the Password Synchronizer:

For WebSphere MQ 5.3:

- com.ibm.mqjms.jar
- com.ibm.mq.jar
- jms.jar
- connector.jar

For WebSphere MQ 6.0:

- com.ibm.mqjms.jar
- com.ibm.mq.jar
- jms.jar
- connector.jar
- dhbcore.jar
- jta.jar

Chapter 11. Log Password Store

The Log Password Store is solely used to log any actions that a normal password store would take. This password store is useful for verifying that the Java Proxy and the native plug-ins are communicating correctly.

Note: This password store logs both usernames and passwords in the log file of the Java Proxy. This password store should be used only for testing purposes, for example during plug-ins configuration and development.

The class for this password store is: `com.ibm.di.plugin.pwstore.log.LogPasswordStore`.

Chapter 12. Troubleshooting problems with the Password Synchronizers

To diagnose problems with the Password Synchronizer, inspect the logs of the plug-in and the Java Proxy components. Each message has an associated timestamp and severity level (error, info and so forth) - use them as hints.

Troubleshooting problems with the plug-in

Consult the section on the specific Password Synchronizer for information where the log messages of the plug-in component are written to. Depending on the plug in this can be a log file, the UNIX syslog, the LDAP server log, and so forth.

1. Check the initialization status.

   Each Password Synchronizer logs a status message on initialization:

   - Verify that the log exists (this is relevant only for Plug-ins that log into a file).
   - Verify that the log contains a message of successful initialization.
   - Verify that the timestamp of the initialization message is recent. It is possible that there are some messages left in the log from previous run of the Plug-in.

   If there is no recent initialization status in the log, this means that the plug-in is not running or failed on initialization before it can write to its log. Possible causes are:

   - The plug-in is not registered correctly into the target system. Review the registration steps from the section the given Password Synchronizer.
   - The plug-in cannot find its configuration file (pwsync.props). Consult the section on the given Password Synchronizer for information how to specify the configuration file to the plug-in.
2. Check for execution errors.

All problems that occur during the operation of the plug-in are logged as error messages.

Troubleshooting problems with the Java Proxy

The Java Proxy component (including the Password Store component) of the Password Synchronizer logs messages into the file specified by the **javaLogFile** configuration parameter (default is proxy.log).

1. 1. Check the initialization status.

   The plug-in starts the Java Proxy on initialization. When the Java Proxy starts it logs a status message.

   - Verify that the log exists (proxy.log).
   - Verify that the log contains a message of successful initialization.
   - Verify that the timestamp of the initialization message is recent. It is possible that there are some messages left in the log from previous run of the Proxy.

   If there is no recent initialization status in the log, this means that the Proxy is not running, or failed on initialization before it can write to its log. A general JavaProxy standard/error log is available in the file `*authentication_folder*/proxy.stdout.log`. The *authentication_folder* is the folder containing the provided plugin configuration file (pwsync.props). If the content of that log file has a message like this: `java.lang.NoClassDefFoundException` then one of the following might be the reason:

- The class-path of the Proxy is incomplete:

  Verify that all third-party libraries needed to run the Password Store are added to the plugins jars folder (the default is to *TDI_Install_dir*/pwd_plugins/jars/). Keep in mind that if the classpath generated by the startProxy script is longer than the length of a shell command allowed by your OS then the JavaProxy might not be able to run.
- Errors when running the startProxy.bat(sh) manually:

  Verify that the startProxy script executes correctly from a command prompt.

  Linux/UNIX Users: Make sure that the configured in the scripts paths are valid for your OS environment.

  Windows Users: The startup scripts of the plugins use JScript to gather the classpath. Make sure that the "Windows Script Host" (WSH) is enabled. If the WSH is disabled on your machine you will receive the following message:

  ```
  Windows Script Host access is disabled on this machine. Contact your administrator for details.
  ```

  when double-clicking on the *TDI_Install_dir*/pwd_plugins/bin/worker.js file.
- An unexpected error occurred on initialization:

  Check the log of the plug-in for error messages related to the startup of the Java Proxy process.

  Locate the command string used by the plug-in to start the Java Proxy (in the plug-in log) and try to execute it in a command shell.

2. Check for execution errors.

   All problems that occur during the operation of the Proxy are logged as error messages.

Chapter 13. Tivoli Identity Manager Integration

This chapter describes the configuration of the Tivoli Identity Manager Integration for the Sun Directory Server Password Synchronizer, IBM Directory Server Password Synchronizer, Windows Password Synchronizer and Password Synchronizer for UNIX and Linux.

This chapter contains the following sections:

Overview

The Tivoli Identity Manager Integration for the Password Sychronizers allows synchronized passwords to be verified by a Tivoli Identity Manager Server's Password Strength Servlet prior to synchronization. This allows Password Synchronization to incorporate password complexity checking via Tivoli Identity Manager Password Policies.

The Tivoli Identity Manager Integration is enabled by utilizing one of the Tivoli Identity Manager Decorator Password Synchronizer classes:

- com.ibm.di.plugin.pwstore.ldap.LDAPPasswordSynchronizerITIMDecorator
- com.ibm.di.plugin.pwstore.jms.MQePasswordStoreITIMDecorator
- com.ibm.di.plugin.pwstore.log.LogPasswordStoreITIMDecorator

Note: The com.ibm.di.plugin.pwstore.log.LogPasswordStoreITIMDecorator password store logs both usernames and passwords in the Java Proxy's log file. This password store should be used only for testing purposes, for example during the plug-ins deployment testing.

Supported Synchronizers

The Tivoli Identity Manager Password Synchronizer Decorator classes are supported by the following Password Synchronizers:

- Password Synchronizer for Windows
- Password Synchronizer for IBM Tivoli Directory Server Synchronizer
- Password Synchronizer for Sun Directory Server
- Password Synchronizer for UNIX and Linux

Note: The Domino HTTP Password Synchronizer does not support integration with ITIM. Custom Password Policies can be created on the Domino Server. Using those Password Policies the passwords can be validated before they are stored.

ITIM Password Strength Validation Communication

External applications that wish to request a password strength validation from ITIM server must create an XML request, and send via HTTPS a servlet hosted by the ITIM server. A sample XML request for password strength validation is shown below:

```
<PSWD_REQ_MSG>
 <CREDENTIALS principal="",pswd="" />
 <REQUEST op="check", srcDN="", userDN="", pswd="" />
</PSWD_REQ_MSG>
```

Credentials Tag:

The credentials represent the user name and password of an ITIM principal. The principal and pswd values are used to enable a client (that is, password store decorator) to authenticate with the ITIM server. The principal must exist in ITIM server, and be given authority to perform the password "check". These credential values will be given to the TDI client component via configuration properties.

Request Tag:

The element attributes are described below. *(This content is taken from "Access Manager 5.1 Password Synchronization between TAM 5.1 and ITIM 4.5")*:

- op – The operation to be performed. This will always be "check", however, "synch" can be used to synchronize the password with ITIM.
- srcDN - Holds the pseudo distinguished name of the service (resource) that is the source of the password strength check. The distinguished name is in the format &<service RDN>,<bu RDN>,<org RDN>,<tenant DN>. An rdn is in the format of attribute=value. The service rdn uniquely identifies a service within a branch of the org-chart. The bu rdn uniquely identifies a container in the org-chart within another branch of the org-chart. There may be 0 or several bu rdn's depending on the org-chart structure. The org rdn uniquely identifies the organization (within a tenant). The tenant dn is the physical distinguished name of the tenant. The following example distinguished name identifies a service named "Test" within the "IT" organizational unit within the "Acme" organization:

  ```
  erservicename=Test,ou=IT,o=Acme,ou=Acme,dc=com
  ```

  ou=Acme,dc=com is the physical dn of the tenant, or the root branch of the directory server in a single-tenant deployment.
- userDN – Holds the Distinguished name of the user (account) within the scope of the source service. For example, the distinguished name of the UNIX user with user id jdoe is eruid=jdoe.
- pswd – Holds the password value to check the strength of.

Configuring Password Synchronizers for Tivoli Identity Manager Integration

Configure the Password Synchronizer to utilize a Tivoli Identity Manager Decorator by setting the **syncClass** property value within the general configuration file (`pwsync.props`) to one of the Decorator classnames shown below:

- com.ibm.di.plugin.pwstore.ldap.LDAPPasswordStoreITIMDecorator
- com.ibm.di.plugin.pwstore.ldap.JMSPasswordStoreITIMDecorator
- com.ibm.di.plugin.pwstore.log.LogPasswordStoreITIMDecorator

The `pwsync.props` file has a section that configures the ITIM integration; specify the following required properties (property names are case-sensitive):

itimPasswordUrl
URL of the Tivoli Identity Manager hosted Password Strength Servlet. For example:

```
https://host:port/passwordsynch/synch
```

itimPrincipalName
Tivoli Identity Manager user name permitted to perform a password check.

itimPrincipalPassword
The password for the Tivoli Identity Manager user name specified in **itimPrincipalName**.

itimSourceDN
The Tivoli Identity Manager service name against which the password check should be performed. For example:

```
erservicename=TDIPasswordService, o=IBM, ou=IBM, dc=com
```

Known Issues

Be aware of the following issues which are known to exist at time of publication:

- When Tivoli Identity Manager Integration is enabled **checkRepository** must be set to *true* in the Password Synchronizer Configuration file.

Appendix A. AIX 5.3 PAM Configuration

AIX 5.3 natively supports the Pluggable Authentication Module (PAM) Framework; Post-installation AIX is configured to utilize Standard Authentication rather than PAM.

The following steps outline how to enable PAM on AIX 5.3:

1. To enable PAM open the /etc/security/login.cfg file and change the value of the **auth_type** property within the **usw** stanza to *PAM_AUTH*. For example:
   ```
   usw:
     shells = ...
     maxlogins = 32767
     logintimeout = 60
     auth_type = PAM_AUTH
   ```
2. Copy the PAM Module to the system.
3. Place the 32-bit version of the PAM Module into the /usr/lib/security directory.
4. Place the 64-bit version of the PAM Module into the /usr/lib/security/64 directory.
5. Ensure that the 32-bit and 64-bit Module share the same file name. For example:
   - /usr/lib/security/libpamtivoli.so
   - /usr/lib/security/64/libpamtivoli.so

   Note: Only use the filename of the modules (both should have the same filenames), not a fully quantified path to the module for the **module_path** section of the Password Synchronizer PAM Module entry. When the PAM Framework loads your module it will either use the 32-bit or 64-bit version, depending on the Operating System mode. If the Operating System is in 32-bit mode then the path to the Password Synchronizer PAM module (pamlibtivoli.so) will be resolved to /usr/lib/security/pamlibtivoli.so. However if the Operating System is in 64-bit mode, the path to the module will be resolved to /usr/lib/security/64/pamlibtivoli.so. Specifying an absolute path to the Password Synchronizer PAM Module binds the PAM Framework to one specific mode only (32-bit or 64-bit).
6. Set the permissions and ownership of the 32-bit and 64-bit modules as follows:
   ```
   chmod 555 /usr/lib/security/libpamtivoli.so
   chown root:system /usr/lib/security/libpamtivoli.so
   chmod 555 /usr/lib/security/64/libpamtivoli.so
   chown root:system /usr/lib/security/64/libpamtivoli.so
   ```
7. Update the /etc/pam.conf file to include the PAM Password Synchronizer Module in the Password Management stack. The following excerpt from the modified AIX 5.3 PAM Framework configuration file shows how your configuration should look:
   ```
   login password  required /usr/lib/security/pam_aix
   passwd password  required libpamtivoli.so use_first_pass
       /opt/IBM/TDI/V7.1/pam/pwsync.props
   rlogin password  required /usr/lib/security/pam_aix
   su password  required /usr/lib/security/pam_aix
   telnet password  required /usr/lib/security/pam_aix
   OTHER password  required /usr/lib/security/pam_prohibit
   ```

8. Test the PAM Framework has been successfully enabled by changing a user's password, if PAM has been successfully configured executing the commands below should cause the user's password to be synchronized to the Password Store.

   ```
   useradd testuser
   passwd testuser
   telnet localhost
   ```

Appendix B. IBM Software Support

IBM Software Support provides assistance with product defects.

Before contacting IBM Software Support, your company must have an active IBM software maintenance contract, and you must be authorized to submit problems to IBM. The type of software maintenance contract that you need depends on the type of product you have:

- For IBM distributed software products (including, but not limited to, Tivoli, Lotus, and Rational® products, as well as DB2® and WebSphere products that run on Windows or UNIX operating systems), enroll in Passport Advantage® in one of the following ways:
  - **Online**: Go to the following Passport Advantage Web page and click **How to Enroll**:

    http://www.lotus.com/services/passport.nsf/WebDocs/Passport_Advantage_Home
  - **By phone**: For the phone number to call in your country, go to the IBM Software Support Web site (http://techsupport.services.ibm.com/guides/contacts.html) and click the name of your geographic region.
- For IBM eServer™ software products (including, but not limited to, DB2 and WebSphere products that run in zSeries®, pSeries®, and iSeries® environments), you can purchase a software maintenance agreement by working directly with an IBM sales representative or an IBM Business Partner. For more information about support for eServer software products, go to the IBM Technical Support Advantage Web page (http://www.ibm.com/servers/eserver/techsupport.html).

If you are not sure what type of software maintenance contract you need, call 1-800-IBMSERV (1-800-426-7378) in the United States or, from other countries, go to the contacts page of the IBM Software Support Handbook on the Web (http://techsupport.services.ibm.com/guides/contacts.html) and click the name of your geographic region for phone numbers of people who provide support for your location.

Follow the steps in this topic to contact IBM Software Support:

1. "Determine the business impact of your problem"
2. "Describe your problem and gather background information" on page 90
3. "Submit your problem to IBM Software Support" on page 90

Determine the business impact of your problem

When you report a problem to IBM, you are asked to supply a severity level. Therefore, you need to understand and assess the business impact of the problem you are reporting. Use the following criteria:

Severity 1	**Critical** business impact: You are unable to use the program, resulting in a critical impact on operations. This condition requires an immediate solution.
Severity 2	**Significant** business impact: The program is usable but is severely limited.

Severity 3	**Some** business impact: The program is usable with less significant features (not critical to operations) unavailable.
Severity 4	**Minimal** business impact: The problem causes little impact on operations, or a reasonable circumvention to the problem has been implemented.

Describe your problem and gather background information

When explaining a problem to IBM, be as specific as possible. Include all relevant background information so that IBM Software Support specialists can help you solve the problem efficiently. To save time, know the answers to these questions:

- What software versions were you running when the problem occurred?
- Do you have logs, traces, and messages that are related to the problem symptoms? IBM Software Support is likely to ask for this information.
- Can the problem be recreated? If so, what steps led to the failure?
- Have any changes been made to the system? (For example, hardware, operating system, networking software, and so on.)
- Are you currently using a workaround for this problem? If so, please be prepared to explain it when you report the problem.

Submit your problem to IBM Software Support

You can submit your problem in one of two ways:

- **Online**: Go to the "Submit and track problems" page on the IBM Software Support site (http://www.ibm.com/software/support/probsub.html). Enter your information into the appropriate problem submission tool.
- **By phone**: For the phone number to call in your country, go to the contacts page of the IBM Software Support Handbook on the Web (http://techsupport.services.ibm.com/guides/contacts.html) and click the name of your geographic region.

If the problem you submit is for a software defect or for missing or inaccurate documentation, IBM Software Support creates an Authorized Program Analysis Report (APAR). The APAR describes the problem in detail. Whenever possible, IBM Software Support provides a workaround for you to implement until the APAR is resolved and a fix is delivered. IBM publishes resolved APARs on the IBM product support Web pages daily, so that other users who experience the same problem can benefit from the same resolutions.

For more information about problem resolution, see “Searching knowledge bases” and “Obtaining fixes” on page 91.

Searching knowledge bases

If you have a problem with your IBM software, you want it resolved quickly. Begin by searching the available knowledge bases to determine whether the resolution to your problem is already documented.

Search the information center on your local system or network

IBM provides extensive documentation that can be installed on your local machine or on an intranet server. You can use the search function of this information center to query conceptual information, instructions for completing tasks, reference information, and support documents.

Search the Internet

If you cannot find an answer to your question in the information center, search the Internet for the latest, most complete information that might help you resolve your problem. To search multiple Internet resources for your product, expand the product folder in the navigation frame to the left and select **Support on the Web**. From this topic, you can search a variety of resources including:

- IBM technotes
- IBM downloads
- IBM Redbooks®
- IBM DeveloperWorks
- Forums and newsgroups
- Google

Obtaining fixes

A product fix might be available to resolve your problem. You can determine what fixes are available for your IBM software product by checking the product support Web site:

1. Go to the IBM Software Support Web site (http://www.ibm.com/software/support).
2. Under **Products A - Z**, select your product name. This opens a product-specific support site.
3. Under **Self help**, follow the link to **All Updates**, where you will find a list of fixes, fix packs, and other service updates for your product. For tips on refining your search, click **Search tips**.
4. Click the name of a fix to read the description and optionally download the fix.

To receive weekly e-mail notifications about fixes and other news about IBM products, follow these steps:

1. From the support page for any IBM product, click **My support** in the upper-right corner of the page.
2. If you have already registered, skip to the next step. If you have not registered, click register in the upper-right corner of the support page to establish your user ID and password.
3. Sign in to **My support**.
4. On the My support page, click **Edit profiles** in the left navigation pane, and scroll to **Select Mail Preferences**. Select a product family and check the appropriate boxes for the type of information you want.
5. Click **Submit**.
6. For e-mail notification for other products, repeat Steps 4 and 5.

For more information about types of fixes, see the *Software Support Handbook* (http://techsupport.services.ibm.com/guides/handbook.html).

Appendix C. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE: This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

AIX
DB2
IBM
IBM(logo)
SecureWay
Tivoli
Tivoli (logo)
Universal Database
WebSphere
IBM-Lotus Trademarks
Domino
iNotes
Lotus Notes
Lotus
Notes

Microsoft, Windows, Windows NT®, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries, or both.

ActionMedia, LANDesk, MMX, Pentium® and ProShare are trademarks of Intel® Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

SET and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.

Program Number: 5724-K74

Printed in USA

SC27-2708-00