# Novell
# Credential Provisioning Policies

**Novell**®

## Novell Trademarks

For Novell trademarks, see the Novell Trademark and Service Mark list (http://www.novell.com/company/legal/trademarks/tmlist.html).

## Third-Party Materials

All third-party trademarks are the property of their respective owners.

# Contents

# About This Guide

Novell® Credential Provisioning policies for Identity Manager 3.5.1 enhance the user provisioning abilities of any Identity Manager driver by providing the capability to simultaneously provision application credentials to the Novell SecretStore® and Novell SecureLogin credential repositories. Additionally, you can provision the SecureLogin passphrase question and answer in environments where non-repudiation is desired.

This guide provides a detailed reference of how to implement Credential Provisioning policies with SecureLogin and SecretStore. The guide does not contain configuration information for Identity Manager, SecureLogin, or SecretStore.

## Audience

This guide is intended for Identity Manager administrators.

## Feedback

We want to hear your comments and suggestions about this manual and the other documentation included with this product. Please use the User Comments feature at the bottom of each page of the online documentation, or go to www.novell.com/documentation/feedback.html and enter your comments there.

## Documentation Updates

For the most recent version of *Credential Provisioning Policies*, visit the Identity Manager Web site (http://www.novell.com/documentation/idm).

## Additional Documentation

For documentation on Identity Manager, see the Identity Manager Documentation Web site (http://www.novell.com/documentation/idm).

For documentation on Identity Manager drivers, see the Identity Manager Documentation Web site (http://www.novell.com/documentation/lg/dirxmldrivers/index.html).

For documentation on SecureLogin, see the Novell SecureLogin Documentation Web site (http://www.novell.com/documentation/secretstore33/index.html).

For documentation on SecretStore, see the Novell SecretStore Documentation Web site (http://www.novell.com/documentation/securelogin60/index.html).

**Documentation Conventions**

In Novell documentation, a greater-than symbol (>) is used to separate actions within a step and items in a cross-reference path.

A trademark symbol ($^{®}$, ™, etc.) denotes a Novell trademark. An asterisk (*) denotes a third-party trademark.

When a single pathname can be written with a backslash for some platforms or a forward slash for other platforms, the pathname is presented with a backslash. Users of platforms that require a forward slash, such as Linux* or UNIX*, should use forward slashes as required by your software.

# Overview

1

Novell® Credential Provisioning policies for Identity Manager enhance the user provisioning abilities of any Identity Manager driver by providing the capability to simultaneously provision application credentials to the Novell SecureLogin® and Novell SecretStore® credential repositories. Additionally, it can provision the SecureLogin passphrase question and answer in environments where non-repudiation is desired.

These features enhance the user Single Sign-On experience and increase the return on investment of Single Sign-On technologies by eliminating the initial setup of SecureLogin account information, providing additional security to application credentials, and reducing the replication of effort normally associated with provisioning Single Sign-On credential stores for users. In addition, the Credential Provisioning policies can use Identity Manager policies to automatically de-provision application credentials to prevent access to application data.

# Novell Credential Provisioning Policies with Novell SecureLogin

# 2

Novell® Credential Provisioning policies allow you to automatically provision application credentials that Novell SecureLogin® supports. This section documents the steps required to configure objects and policies in Identity Manager. It does not contain deployment and configuration information for any SecureLogin components. For SecureLogin documentation, see the Novell SecureLogin 6.0 Documentation Web Site (http://www.novell.com/documentation/securelogin60/index.html).

To implement Credential Provisioning with SecureLogin requires a repository object, an application object, and policies. The repository and application objects store the SecureLogin information so that Identity Manager can use it. The policies are used to enable a driver to use Credential Provisioning. See Chapter 3, "Implementing Novell Credential Provisioning Policies with Novell SecureLogin," on page 15 for more information.

You can also configure the following options:

- Credential Provisioning can be provided by the Publisher channel, Subscriber channel, or both channels.
- SecureLogin synchronization can occur as part of an application password synchronization or can be triggered by some other event.
- Web Services credentials can be provisioned without provisioning accounts for the application.
- An initial SecureLogin passphrase question and answer can be provisioned.

You can use random password generation to set the passwords for user accounts on connected systems to further secure your Identity Management environment. For more information, see *Novell Identity Manager 3.5.1 Administration Guide* for using random password generation.

Figure 2-1 on page 12 shows a typical, yet simple, scenario involving the provisioning of the SecureLogin credentials for a new User of a SAP* Finance application in a Finance department. SAP User provisioning is used for this example because it is an application that requires more login parameters than the typical username and password provided for most applications.

This department provisions new users into the Identity Vault via a SAP HR system and Identity Manager. Depending on organizational information, the User object is then provisioned into a department authentication tree implemented on Active Directory*. This is where new users authenticate to the network and is therefore the location for the SecureLogin credential repository. As users are subsequently provisioned by Identity Manager to the various finance applications, their credentials for those systems are synchronized to the SecureLogin store in Active Directory.

Figure 2-1 shows user Glen's authentication credentials being provisioned. When Glen authenticates to his department's Active Directory authentication domain and launches the SecureLogin client, he has single sign-on to his SAP Finance account without ever needing to enter, or even know, his password on that system.

*Figure 2-1*   *Credential Provisioning with SecureLogin*
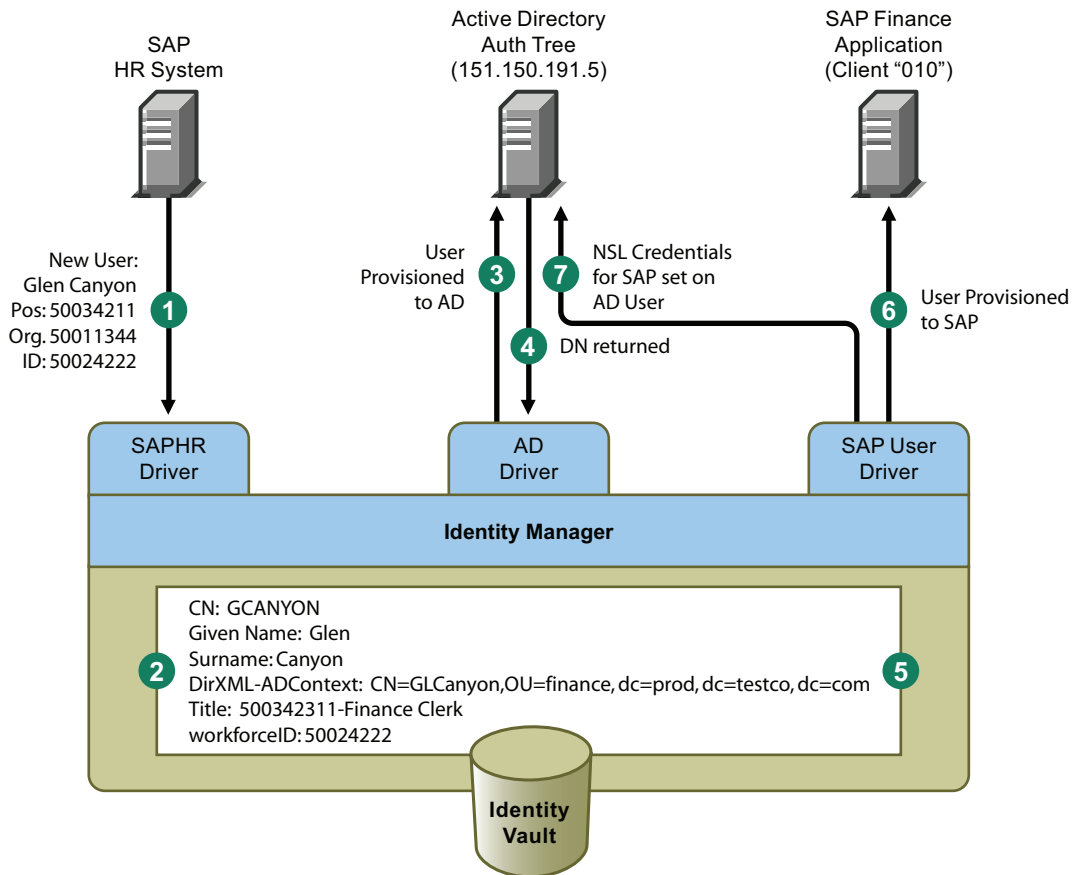


Figure 2-1 illustrates the following steps:

1.  A SAP HR system publishes the data for a newly hired user named Glen Canyon. The Identity Manager SAP HR driver processes this data.

2.  A new User object is created in the Identity Vault with a CN value of GCANYON and a workforceID value of 50024222. Because this user is assigned to the Finance organization of his company, he needs to authenticate to the Finance department Active Directory server in the finance.prod.testco.com domain. The Identity Manager Active Directory driver that synchronizes that domain now uses the Identity Vault information.

3.  Glen is provisioned to the Finance department Active Directory server.

4.  The driver is configured to obtain Glen's fully distinguished LDAP name: CN=GLCanyon,OU=finace,dc=prod,dc=testco,dc=com.

5.  The driver places the name into the DirXML-ADContext attribute of the GCANYON user in the Identity Vault.

    Now that the required attributes are available in the Identity Vault, the SAP User Management driver processes the attributes of the GCANYON object.

6.  Because Glen is in the Finance organization, the driver provisions a SAP user account GCANYON on the SAP Finance server.

7. After the account creation is successful, the SAP User Management driver policies provision Glen's SAP authentication credentials to his AD user account. Because the command is an Add operation, the policies also provision his SecureLogin passphrase question and answer.

# Implementing Novell Credential Provisioning Policies with Novell SecureLogin

3

The implementation of Novell* Credential Provisioning policies with Novell SecureLogin is very customizable. The steps to implement it are different depending upon the platforms SecureLogin is installed on, the applications that are provisioned, and which Identity Manager drivers are involved.

To implement Credential Provisioning policies with SecureLogin, see the following topics:

## 3.1 Requirements for Credential Provisioning Policies with Novell SecureLogin

In order to use Credential Provisioning Policies with SecureLogin, the following must be in place:

- Identity Manager 3.0.1 or above
- eDirectory™ 8.7x or eDirectory 8.8.1 or above; eDirectory 8.8 is not supported.
- Verify that `jsso.jar`, `idmcp.jar`, and `jnet.jar` are in the standard location for Identity Manager Java libraries.
- Novell SecureLogin 6.0 or above

After you have verified that your environment meets the requirements, proceed to Section 3.2, "Extending the LDAP Schema for Novell SecureLogin," on page 16.

## 3.2  Extending the LDAP Schema for Novell SecureLogin

When SecureLogin is deployed on eDirectory servers, a tool called `ndsschema.exe` is utilized to extend the eDirectory schema with a set of SecureLogin attributes that are used to store encrypted credentials, policies, etc. on Users and container objects. These attributes are:

- Prot:SSO Auth
- Prot:SSO Entry
- Prot:SSO Entry Checksum
- Prot:SSO Profile
- Prot:SSO Security Prefs
- Prot:SSO Security Prefs Checksum

These attributes are specific to eDirectory and are required in order for the SecureLogin product to function. The provisioning API provided in Identity Manager 3.0 Support Pack 1 utilizes the LDAP namespace to perform its functions so that it can work with any SecureLogin credential store.

In order to provide LDAP mappings to the attributes listed above, a second tool provided with the SecureLogin product must be utilized. The tool name is `ldapschema.exe`, and it is used in eDirectory environments to provide the LDAP namespace mapping to the eDirectory attributes.

See Preparing for an LDAP Directory (http://www.novell.com/documentation/securelogin60/index.html?page=/documentation/securelogin60/nslinst/data/b1tnxsu.html#b1tnxsu) in the *Novell SecureLogin 6.0 Installation Guide*.

After running `ldapschema.exe`, verify the mappings by checking the LDAP Group attribute map in iManager.

1  In iManager, click *LDAP > LDAP Options*.

2  Select the LDAP Group associated with your eDirectory servers that host SecureLogin.

3  From the LDAP Group properties page, select the *Attribute Map* option and verify that the eDirectory attributes are correctly mapped:

| eDirectory Attributes | LDAP Attributes |
| --- | --- |
| Prot:SSO Auth | protocom-SSO-Auth-Data |
| Prot:SSO Entry | protocom-SSO-Entries |
| Prot:SSO Entry Checksum | protocom-SSO-Entries-Checksum |
| Prot:SSO Profile | protocom-SSO-Profile |
| Prot:SSO Security Prefs | protocom-SSO-Security-Prefs |
| Prot:SSO Security Prefs Checksum | protocom-SSO-Security-Prefs-Checksum |

4  After the schema is extended, proceed to Section 3.3, "Determining Deployment Configuration Parameters for Novell SecureLogin," on page 17.

# 3.3 Determining Deployment Configuration Parameters for Novell SecureLogin

In order to provide the synchronization functionality described in the deployment scenario illustrated in Figure 2-1, the first step is to gather all of the business process information related to the Identity Manager and SecureLogin environments. You can print Table 3-1, "Credential Provisioning Policies Worksheet for SecureLogin," on page 17, and use it as a worksheet to record the information.

*Table 3-1   Credential Provisioning Policies Worksheet for SecureLogin*

| Configuration Information Needed | Information |
| --- | --- |
| 1) Which applications will be configured for SecureLogin Single Sign-On provisioning? | |
| 2) Verify that SecureLogin application definitions are preconfigured on the authentication server and are inheritable by new users provisioned to those systems. | |
| 3) The DNS name or IP address of the SecureLogin repository server. | |
| 4) The SSL LDAP port for the SecureLogin repository server. | |
| 5) The fully qualified LDAP distinguished name of the administrator for the SecureLogin repository server. | |
| 6) The password of the administrator for the SecureLogin repository server. | |
| 7) The full path and the name of the SSL certificate exported from the SecureLogin server. The certificate must be local to the Identity Manager server. | |
| 8) Determine if one SecureLogin repository will be used by multiple drivers or if each driver will use a separate repository. | |
| 9) The application ID for each SecureLogin application. | |
| 10) List all required authentication keys for each application, such as, Username, Password, Client, and Language. They might be different for each application. | |
| 11) Determine if any of the authentication key values can be set with a static value. | |
| 12) For non-static values that are or can be different for each user, make a note of the source of the non-static information (event information or Identity Vault attribute values). | |

| Configuration Information Needed | Information |
|---|---|
| 13) If you are implementing SecureLogin provisioning on a driver that is also synchronizing a password to the target application, determine if the SecureLogin provisioning takes place before or after the password is set in the target application server. | |
| 14) The name of the Driver object where the repository and application objects are to be stored. (Can be different drivers.) | |
| 15) Determine the DN of the User objects for the target application. | |
| 16) If you are implementing a SecureLogin passphrase, determine the passphrase question and answer. | Question: Answer: |

## 3.3.1 Example Provisioning Configuration Data

Using the provisioning scenario in , the following example data provisions a user's SecureLogin credentials for the SAP Finance server for users in the Finance Active Directory authentication tree:

*Table 3-2*  *Example Credential Provisioning Policies Worksheet for SecureLogin*

| Configuration Information Needed | Information |
|---|---|
| 1) Which applications will be configured for SecureLogin Single Sign-On provisioning? | SAP Finance Application |
| 2) Verify that SecureLogin application definitions are preconfigured on the authentication server and are inheritable by new users provisioned to those systems. | Verified |
| 3) The DNS name or IP address of the SecureLogin repository server. | 151.150.191.5 |
| 4) The SSL LDAP port for the SecureLogin repository server. | 636 |
| 5) The fully qualified LDAP distinguished name of the administrator for the SecureLogin repository server. | cn=admin,ou=prod,dc=testco,dc=.com |
| 6) The password of the administrator for the SecureLogin repository server. | dixml |
| 7) The full path and the name of the SSL certificate exported from the SecureLogin server. The certificate must be local to the Identity Manager server. | `c:\novell\nds\FinanceAD.cer` |

| Configuration Information Needed | Information |
|---|---|
| 8) Determine if one SecureLogin repository will be used by multiple drivers or if each driver will use a separate repository. | For this example, there is only one repository. |
| 9) The application ID for each SecureLogin application. | SAP - 151.150.191.27 |
| 10) List all required authentication keys for each application, such as, Username, Password, Client, and Language. They might be different for each application. | SAP Client 010 Login Parameter Client<br>SAP Client 010 Login Parameter Language<br>SAP Client 010 Login Parameter Username<br>SAP Client 010 Login Parameter Password |
| 11) Determine if any of the authentication key values can be set with a static value. | SAP Client 010 Login Parameter Client:"010"<br>SAP Client 010 Login Parameter Language: "EN" |
| 12) For non-static values that are or can be different for each user, make a note of the source of the non-static information (event information or Identity Vault attribute values). | SAP Client 010 Login Parameter Username: Identity Vault attribute "sapUsername"<br>SAP Client 010 Login Parameter Password: Event `<password>` |
| 13) If you are implementing SecureLogin provisioning on a driver that is also synchronizing a password to the target application, determine if the SecureLogin provisioning takes place before or after the password is set in the target application server. | After |
| 14) The name of the Driver object where the repository and application objects are to be stored. (Can be different drivers.) | SAP driver |
| 15) Determine the DN of the User objects for the target application. | Identity Vault attribute "DirXML-ADContext" |
| 16) If you are going to provision the SecureLogin passphrase, determine the passphrase question and answer. | Question: "Employee code?"<br>Answer: Identity Vault attribute "workforceID" |

Miscellaneous Environment Information:

- The Finance department AD tree serves as the SecureLogin repository for all Finance applications.
- All finance department provisioning drivers are in a driver set called Finance Drivers.
- The SAP user account must be deleted and the SecureLogin credentials for the SAP user account must be removed from the Active Directory user when the Identity Vault attribute "employeeStatus" is set to the value "I".

After all of the configuration data has been determined, proceed to .

# 3.4  Creating a Repository Object for Novell SecureLogin

Repository objects store static configuration information for SecureLogin. Repository information is independent from the applications that consume the application credentials. This information is

applicable for all provisioning events regardless of the connected system (for example SAP, PeopleSoft*, Notes*, etc.). The repository object can be created in Designer or iManager.

## 3.4.1  Creating a Repository Object for Novell SecureLogin in Designer

The following is one of many methods you can use to create the repository object in Designer.

**1** Right-click the driver object where you want to store the repository object in the outline view.

**2** Click *New > Credential Repository*.

Credential Application...
Credential Repository...
DirXML Script...
ECMAScript...
Entitlement...
Job...
Mapping Table...
Resource...
Schema Map...
XSLT...

From Copy...
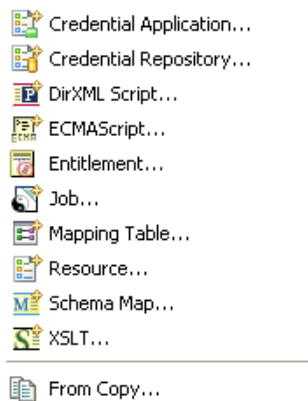
**3** Specify a name for the repository object.

**4** Select *NSLRepository.xml* to use the SecureLogin template.

**Create Repository**

Give a name for the repository object and select the default template to start with

Name: Sap_Repository

⊙ NSLRepository.xml
    Novell SecureLogin 6.0 Repository template

◯ NSSRepository.xml
    Novell SecretStore 3.3 Repository template

☑ Open the editor after creating the object.

Verify that the *Open the editor after creating the object* check box is selected.

**5** Click *OK*.

**6** Click *Yes* to save the new repository object.

**File Conflict**

? Before editing this item, you need to save. Do you want to save this editor's changes and continue?

    Yes    No

**7** Specify the DNS name or IP address of the SecureLogin server. (See worksheet item 3).

SecureLogin Server Name or Address:

151.150.191.5

**8** Specify the SSL port for the SecureLogin server. (See worksheet item 4).

SecureLogin Server SSL Port: 636   ⓘ

**9** Specify the full path to the SSL certificate exported from the SecureLogin server. The path must include the certificate name and must be local to the Identity Manager server. (See worksheet item 7).

SecureLogin Server SSL Certificate Path:

c:\novell\nds\FinanceAD.cer

The SecureLogin server can run on multiple platform types. Refer to the platform-specific documentation for information on how to export the SSL certificates.

**10** Specify the fully qualified LDAP distinguished name of the SecureLogin administrator. (See worksheet item 5).

SecureLogin Administrator:  cn=admin,ou=prod,dc=testco,dc=.com

**11** Click *Set password*.

SecureLogin Administrator Password:  [Set Password...]  [Remove Password]  ⓘ

**12** Specify the SecureLogin administrator's password twice, then click *OK*. (See worksheet item 6).

Set Password

Enter password:

••••••

Re-enter password:

••••••

[ OK ]     [ Cancel ]

**13** Review the information, then click the *Save* icon 🖫 to save the information.

**14** (Optional) If you want to create other configuration parameters for the repository object, click the *Add new item* ✚ icon.

Project 1 - Devel...    project 2 - Devel...    Sap_Repository ✕

**Credential Provisioning**                              ✚  –  ⑦

Sap_Repository.SAP-USER.Driver Set

**Repository Configuration**

ⓘ The following set of configuration parameters are required Connection Parameters specified by the Novell SecureLogin Provisioning API implemented in the class NSLProvision. Failure to provide correct values for ANY of these parameters results in the inability to provision or deprovision SecureLogin credentials from Identity Manager policies.

**14a** Specify a name for the parameter.

**14b** Specify a display name for the parameter.

**14c** Specify a description of the parameter for your reference.

The parameter is stored as a string.



**14d** Click *OK*.

**14e** Click the *Save* icon 🖫 to save the repository object.

After the repository object is created, proceed to .

## 3.4.2 Creating a Repository Object for Novell SecureLogin in iManager

**1** In iManager, select *Credential Provisioning > Configuration.*

**2** Browse to and select the Driver object where the repository object will be stored.

**3** Click *New* to create a repository.

> Repositories  Applications
>
> New... | Delete
>   Name
>
> No repositories were found- Select 'New'

**4** Specify a name for the repository object, then select *NSLRepository.xml* to use the SecureLogin template to create a repository.

> ### Create Repository
> Enter the name for the new repository object.
> Name:
> Sap_Repository
>
> Select the template used to create the new repository object.
> Repository Templates
>   ⦿ NSLRepository.xml
>     Description: Novell SecureLogin 6.0 Repository template
>
>   ○ NSSRepository.xml
>     Description: Novell SecretStore 3.3 Repository template
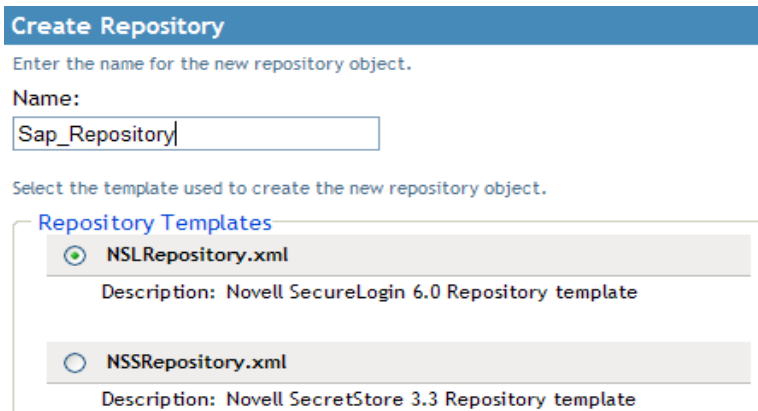
**5** Click *OK*.

**6** Specify the DNS name or IP address of the SecureLogin server. (See worksheet item 3).

> ☐ SecureLogin Server Name or Address ⓘ    151.150.191.5

**7** Specify the SSL port for the SecureLogin server. (See worksheet item 4).

> ☐ SecureLogin Server SSL Port ⓘ    636

**8** Specify the full path to the SSL certificate exported from the SecureLogin server. The path must include the certificate name and must be local to the Identity Manager server. (See worksheet item 7).

> ☐ SecureLogin Server SSL Certificate Path ⓘ   c:\novell\nds\FinanceAD.cer

The SecureLogin server can run on multiple platform types. Refer to the platform-specific documentation for the steps on how to export the SSL certificate.
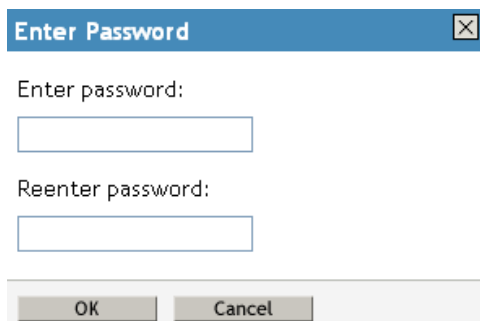
**9** Specify the fully qualified LDAP distinguished name of the SecureLogin administrator. (See worksheet item 5).



**10** Click *Set password*.



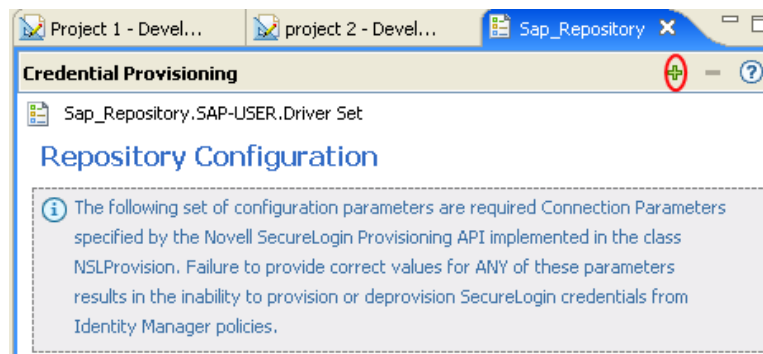**11** Specify the SecureLogin administrator's password twice, then click *OK*. (See worksheet item 6).



**12** Review the values specified, then click *OK*.

**13** (Optional) If you need to create other configuration parameters for the repository, click *New*.



**13a** Specify a name for the parameter.

**13b** Specify a display name for the parameter.

**13c** Specify a description of the parameter for your reference.

The parameter is stored as a string.

**Global Configuration Value Definition**

Global Configuration Values are a means through which the behavior of an Identity Manager driver configuration can be changed without requiring any policy to be changed.

Name:

Display name:

Description:

Type:
string

**13d** Click *OK*.

After the repository object is created, proceed to .

## 3.5 Creating an Application Object for Novell SecureLogin

Application objects store application authentication parameter values for SecureLogin. Application information is specific to the applications that are consuming the application credential (for example, GroupWise® client information or SAP database client information). The application objects can be created in Designer or iManager.

### 3.5.1 Creating an Application Object for Novell SecureLogin in Designer

The following is one of many methods you can use to create the application object in Designer.

**1** In the Outline view, right-click the driver object where you want to store the application object.

**2** Click *New > Credential Application*.



**3** Specify a name for the application object.

**4** Select *NSLApplication.xml* to use the SecureLogin template.



Verify that the *Open the editor after creating the object* check box is selected.

**5** Click *OK*.

**6** Click *Yes* to save the new application object.



**7** Specify the SecureLogin Application ID. (See worksheet item 9).



To find the application ID in SecureLogin, click *My Logins*. The application ID is stored in the *Id* field.



**8** Click the *Save* icon ▤ to save the application.

**9** Click the *Add new item* icon ✚ to add the authentication keys required for the application.
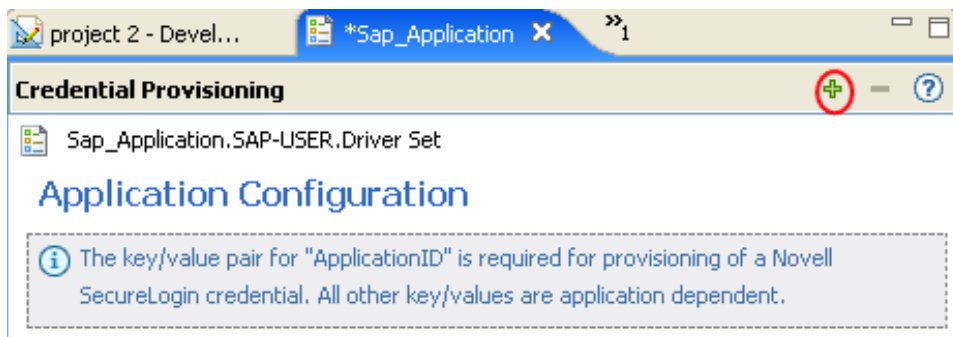


**9a** Specify a name for the authentication key.

**9b** Specify a display name for the authentication key.

**9c** Specify a description of the authentication key for your reference.

The authentication key is stored as a string.



**9d** Click *OK*.

**9e** Repeat Step 9 for each new authentication key that needs to be entered.

To find the authentication key for your application, manually create a SecureLogin credential for a user in the application and have the user log in. After the user has logged in, the authentication key information is displayed under My Logins in the SecureLogin administration window.

**10** Specify the authentication key value if it is a static value that is shared by all user credentials.
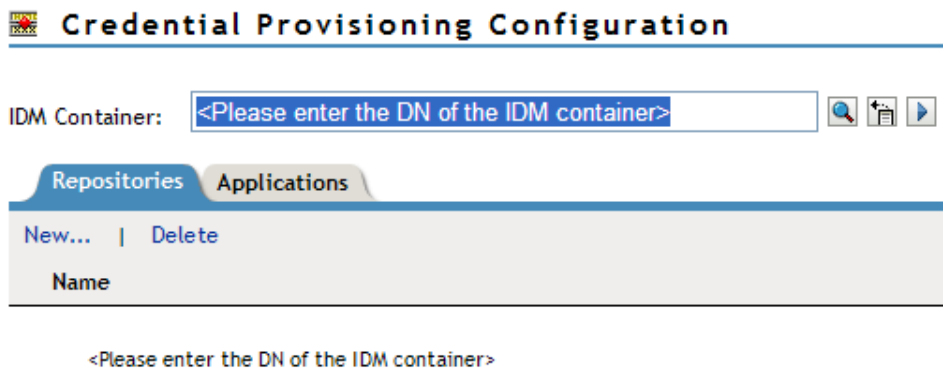
**11** Click the Save icon 🖫 to save the application.

After the application object is created, proceed to "Configuring Credential Provisioning Policies for Novell SecureLogin" on page 32.

### 3.5.2 Creating an Application Object for Novell SecureLogin in iManager

**1** In iManager, select *Credential Provisioning > Configuration*.

**2** Browse to and select the Driver object where the application object will be stored.

**3** Select the *Applications* tab, then click *New*.
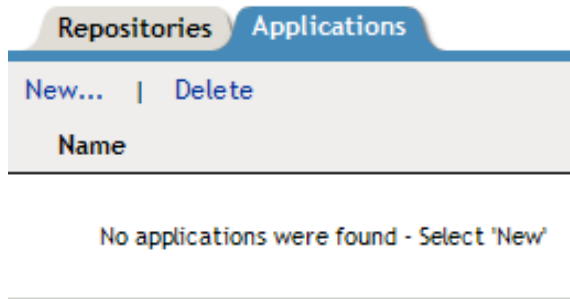


**4** Specify a name for the application object.

**5** Select *NSLApplication.xml* to use the SecureLogin template to create an application.
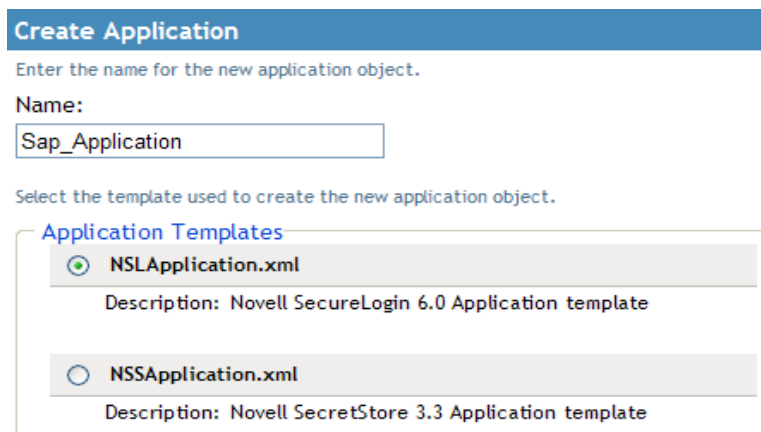


**6** Click *OK*.

**7** Specify the *SecureLogin Application ID*. (See item worksheet 9).



To find the application ID in SecureLogin, click *My Logins*. The application ID is stored in the *Id* field.
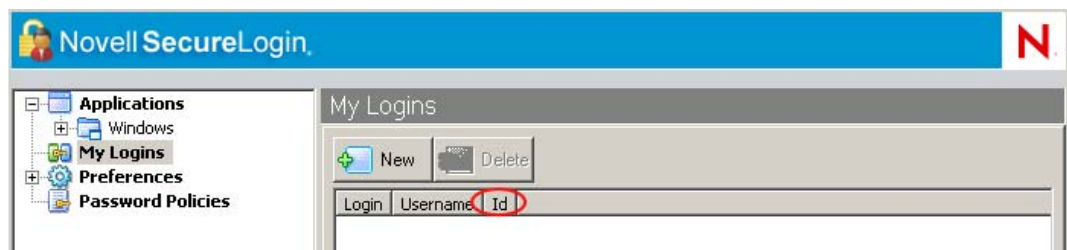
**8** Click *New* to create an authentication key parameter. (See worksheet item 10).

**Credential Provisioning**

**Configuration**

The key/value pair for "ApplicationID" is required for provisioning of a Novell SecureLogin credential. All other key/values are application dependent.

**Application Configuration**

New... | Delete

| Display Name | Value |

**8a** Specify a name for the authentication key.

**8b** Specify a display name for the authentication key.

**8c** Specify a description of the authentication key for your reference.

The authentication key is stored as string.

**Global Configuration Value Definition**

Global Configuration Values are a means through which the behavior of an Identity Manager driver configuration can be changed without requiring any policy to be changed.

Name:

Display name:

Description:

Type:
string

To find the authentication key for your application, manually create a SecureLogin credential for a user in the application and have the user log in. After the user has logged in, the authentication key information is displayed under *My Logins* in the SecureLogin administration window.

**8d** Click *OK*.

**8e** Specify the value of the authentication key, if it is static, then click *OK*.

| Application Configuration | |
| --- | --- |
| New...   \|   Delete | |
| **Display Name** | **Value** |
| ☐ SecureLogin Application ID ⓘ | SAP - 151.150.191.27 |
| ☐ Client ⓘ | 010 |
| ☐ Language ⓘ | EN |
| ☐ Username ⓘ | |
| ☐ Password ⓘ | |

After the application object is created, proceed to "Configuring Credential Provisioning Policies for Novell SecureLogin" on page 32.

# 3.6 Configuring Credential Provisioning Policies for Novell SecureLogin

After the repository and application objects are created, policies need to be created to provision SecureLogin information. The policies can be created in Designer or iManager.

- Section 3.6.1, "Creating Credential Provisioning Policies for Novell SecureLogin in Designer," on page 32
- Section 3.6.2, "Creating Credential Provisioning Policies for Novell SecureLogin in iManager," on page 34

## 3.6.1 Creating Credential Provisioning Policies for Novell SecureLogin in Designer

The policies use the information stored in the repository and application objects.

**1** In the Policy Builder, create a new policy.

**2** (Optional) To clear the SSO credential, so objects can be deprovisioned, select the *clear SSO credential* action, then fill in the following fields:

Do | clear SSO credential | ▼ | ⑦

Specify credential repository object DN: *

☑ Render browsed DN relative to policy

Specify target user DN: *

Populate the following from an application object

Specify application credential ID: *

Specify login parameter strings:

- ◆ **Specify Credential Repository Object DN:** Browse to and select the repository object. (See worksheet item 8).

- ◆ **Specify Target User DN:** Create the DN of the target users by using the Argument Builder. (See worksheet item 15).

- ◆ **Specify Application Credential ID:** Specify the application ID. (See worksheet item 9).

- ◆ **Specify Login Parameter Strings:** Launch the String Builder and enter each authentication key for the application. (See worksheet item 10).

**3** (Optional) To set the SSO credential when a user object is created or when a password is modified, select the *set SSO credential* action, then fill in the following fields:

Do | set SSO credential | ▼ | ⑦

Specify credential repository object DN: *

☑ Render browsed DN relative to policy

Specify target user DN: *

Populate the following from an application object

Specify application credential ID: *

Specify login parameter strings:

- ◆ **Specify Credential Repository Object DN:** Browse to and select the repository object. (See worksheet item 8).

- ◆ **Specify Target User DN:** Create the DN of the target users by using the Argument Builder. (See worksheet item 15).

- ◆ **Specify Application Credential ID:** Specify the application ID. (See worksheet item 9).

- ◆ **Specify Login Parameter Strings:** Launch the String Builder and enter each authentication key for the application. (See worksheet item 10).

**4** (Optional) To create a SecureLogin passphrase and answer for a user object when it is provisioned, select the *set SSO passphrase action*, then fill in the following fields:



- ◆ **Specify Credential Repository Object DN:** Browse to and select the repository object. (See worksheet item 8).

- ◆ **Specify Target User DN:** Create the DN of the target users by using the Argument Builder. (See worksheet item 15).

- ◆ **Question String:** Specify the passphrase question. (See worksheet item 16).

- ◆ **Answer String:** Specify the passphrase answer. (See worksheet item 16).

### 3.6.2  Creating Credential Provisioning Policies for Novell SecureLogin in iManager

The policies use the information stored in the repository and application objects.

**1** In the Policy Builder, create a new policy.

**2** (Optional) To clear the SSO credential, so objects can be deprovisioned, select the *clear SSO credential* action, then fill in the following fields:



- ◆ **Enter Credential Repository Object DN:** Browse to and select the repository object. (See worksheet item 8).

- ◆ **Enter Target User DN:** Create the DN of the target users by using the Argument Builder. (See worksheet item 15).

- ◆ **Enter Application Credential ID:** Specify the application ID. (See worksheet item 9).

- ◆ **Enter Login Parameter Strings:** Launch the String Builder and enter each authentication key for the application. (See worksheet item 10).

**3** (Optional) To set the SSO credential when a user object is created or when a password is modified, select the *set SSO credential* action, then fill in the following fields:



- **Enter Credential Repository Object DN:** Browse to and select the repository object. (See worksheet item 8).
- **Enter Target User DN:** Create the DN of the target users by using the Argument Builder. (See worksheet item 15).
- **Enter Application Credential ID:** Specify the application ID. (See worksheet item 9).
- **Enter Login Parameter Strings:** Launch the String Builder and enter each authentication key for the application. (See worksheet item 10).

**4** (Optional) To create a SecureLogin passphrase and answer for a user object when it is provisioned, select the *set SSO passphrase action*, then fill in the following fields:



- **Enter Credential Repository Object DN:** Browse to and select the repository object. (See worksheet item 8).
- **Enter Target User DN:** Create the DN of the target users by using the Argument Builder. (See worksheet item 15).
- **Enter Question and Answer Strings:** Launch the String Builder and enter the passphrase question and answer. (See worksheet item 16).

## 3.7 Example Credential Provisioning Policies

The provisioning policies can be implemented and customized to meet the needs of your environment. The following example explains how to implement the polices for the scenario presented in Figure 2-1 on page 12.

In the Finance scenario, SecureLogin provisioning occurs after a password is successfully set in SAP. Most of the necessary parameters are statically configured and available to all policies through the repository and application objects. However, there are non-static data parameters (sapUsername, password, DirXML-ADContext, and workforceID) that are available only after the SAP User Management driver `<add>` or `<modify-password>` commands complete and the `<output>`

status document is returned from the SAP User Management driver shim. The `<ouput>` document no longer contains any of the Subscriber channel operation attributes and the user context of the command is lost, thus preventing queries on the object. It is therefore necessary to do the following:

 ◆ Make sure the SAP User driver's Subscriber Create policy enforces the presence of the non-static data parameters.

 ◆ Cache the non-static parameters required for the provisioning operation prior to issuing the Subscriber command to the SAP User driver shim.

 ◆ Retrieve cached data for use in SecureLogin provisioning after the command completes successfully.

---

**NOTE:** Sample policies are available in XML format on the Identity Manager 3.0 Support Pack 1 media. The filenames are `SampleInputTransform.xml`, `SampleSubCommandTransform.xml`, and `SampleSubEventTransform.xml`. The files are found in the following directories, depending upon the platform:

 ◆ `linux\setup\utilities\cred_prov`

 ◆ `nt\dirxml\utilities\cred_prov`

 ◆ `nw\dirxml\utilities\cred_prov`

The files are installed to the Identity Manager server, if Credential Provisioning Sample Policies is selected during the installation of the utilities. The sample policies are installed to the following locations, depending upon the platform:

 ◆ Windows: `C:\Novell\NDS\DirXMLUtilities` (default; the user can change it during install)

 ◆ NetWare®: `SYS:\System\DirXmlUtilities`

 ◆ Linux (eDir 8.7): `/usr/lib/dirxml/rules/credprov`

 ◆ Linux (eDir 8.8.1): `/opt/novell/eDirectory/lib/dirxml/rules/credprov` (default; the user can change it during install)

The sample policies provide a starting point to develop a policy that works for your environment.

---

# 3.8  Operation Data Caching

The mechanism that is available for required operation data caching is the `<operation-data>` element. Because you might need to provision the SecureLogin account from either an `<add>` or `<modify-password>` command, a logical place to implement the non-static data caching policy is in the Subscriber Command Transformation policy. The following example shows a typical SecureLogin Provisioning `<operation-data>` element:

```
<operation-data>
    <nsl-sync-data>
        <nsl-target-user-dn>
cn=GLCANYON,ou=finance,dc=prod,dc=testco,dc=com
        </nsl-target-user-dn>
        <nsl-app-username>GCANYON</nsl-app-username>
        <password><!-- content suppressed --></password>
        <nsl-passphrase-answer>50024222</nsl-passphrase-answer>
```

```
        </nsl-sync-data>
</operation-data>
```

In the sample Finance department scenario from Figure 2-1 on page 12, the following values are needed to populate the operation data payload:

- The `<nsl-target-user-dn>` element is populated with the value of the DirXML-ADContext attribute from the Identity Vault, which was set by the Active Directory driver. To ensure that the SAP User driver is notified when the value is set by the AD driver, make sure you add DirXML-ADContext to the Subscriber filter as a notify attribute.

- The `<nsl-app-username>` element is populated by the value of the sapUsername attribute which, for an `<add>` command, is generated by the Create policy of the SAP User driver and is therefore available as an operation attribute. With the SAP User driver, the SAP User name value is part of the association value. This means that for password modification events the names are parsed from the association.

- The password element is populated with the value of the `<password>` element in the `<add>` or `<modify-password>` command.

- The `<nsl-passphrase-answer>` element is populated with the value of the workforceID attribute from the Identity Vault, which was set by the SAP HR driver. Although this value should be set during initial provisioning to the Identity Vault, it is still a good practice to add workforceID to the Subscriber filter as a notify attribute.

## 3.9  SecureLogin Provisioning

In the provisioning scenario, the first available location from which the operation data can be retrieved and utilized for SecureLogin credential provisioning is in the driver's Input Transformation policy. In the sample scenario, three policies are implemented:

- Set SecureLogin Credentials after successful password synchronization.
- Set SecureLogin Passphrase and Answer
- Remove SecureLogin Credentials if Application User Deleted (Identity Vault object not deleted)

  NOTE: There is a sample policy in the `SampleInputTransform.xml` file that sets SecureLogin credentials after a successful password synchronization occurs. The file is located in the Credential Provisioning folder on the Identity Manager 3.0 Support Pack 1 media.

The Set SecureLogin Credentials policy needs to make sure the provisioning happens only if the returned command status is success and the previously set `<operation-data>` is present.

## 3.10  SecureLogin Deprovisioning

There are many scenarios that can utilize a policy in which a user account for a connected application is deleted and the Identity Vault account remains. In the Finance scenario, there is a requirement to delete the SAP User account and deprovision the SecureLogin credentials when the User's Identity Vault employeeStatus attribute value is set to "I". To handle this situation, the SAP User driver's Subscriber Event Transformation contains a policy to transform the modify attribute value into an object delete. Because the Active Directory account name is still needed after the `<delete>` command is completed, the `<operation-data>` event needs to be set on the

`<delete>` command so it is available to the SecureLogin deprovisioning policy in the Input Transformation policy.

```
<operation-data>
    <nsl-sync-data>
        <nsl-target-user-dn>
cn=GLCANYON,ou=finance,dc=prod,dc=testco,dc=com
        </nsl-targer-user-dn>
    </nsl-sync-data>
</operation-data>
```

The policy for transforming the `<modify>` event into a `<delete>` and creating this element is available in the sample Credential Provisioning policies in the `SampleSubEventTransform.xml` file.

After the policy is created, proceed to .

# Novell Credential Provisioning Policies with Novell SecretStore

4

Novell® Credential Provisioning policies allow you to provision application credentials to User objects in a Novell SecretStore® repository. The capability to provision the Application Server and the User credentials as part of a standard Identity Manager provisioning scenario provides a much more secure and synchronized Web Single Sign-On experience for users.

This section contains the steps required to configure objects and policies in Identity Manager. It does not contain deployment and configuration information for any SecretStore components. For SecretStore documentation, see Novell SecretStore 3.3.3 documentation (http://www.novell.com/documentation/secretstore33/index.html).

To implement Credential Provisioning with SecretStore requires a repository object, an application object, and creating policies. Repository and application objects store the SecretStore information so that Identity Manager can use it. The policies are used so that any driver can be enabled to use Credential Provisioning. It is also possible to configure the following options:

- Credential Provisioning can be provided by the Publisher channel, Subscriber channel, or both channels.
- SecretStore synchronization can occur as part of an application password synchronization or be triggered by some other event.
- Web Services credentials can be provisioned without provisioning accounts for the application.

You can use random password generation to set the passwords for user accounts on connected systems to further secure your Identity Management environment. For more information, see the *Novell Identity Manager 3.5.1 Administration Guide* for using random password generation.

Figure 4-1 shows a typical, yet simple, scenario involving the provisioning of the Single Sign-On credentials for a new user in GroupWise®. This department provisions new users into the Identity Vault via a SAP HR system and Identity Manager. Depending on organizational information, the user is then provisioned into a department authentication tree implemented on eDirectory™. This is where new users authenticate to the network, and is also the repository of GroupWise security credentials that Novell iChain® or Novell Access Manager™ utilizes to provide secure Single Sign-On functionality from outside the company firewall. As users are subsequently provisioned by Identity Manager to GroupWise, the credentials for those systems are synchronized to their SecretStore attributes in the authentication tree.

**Figure 4-1**  *Credential Provisioning with SecretStore*
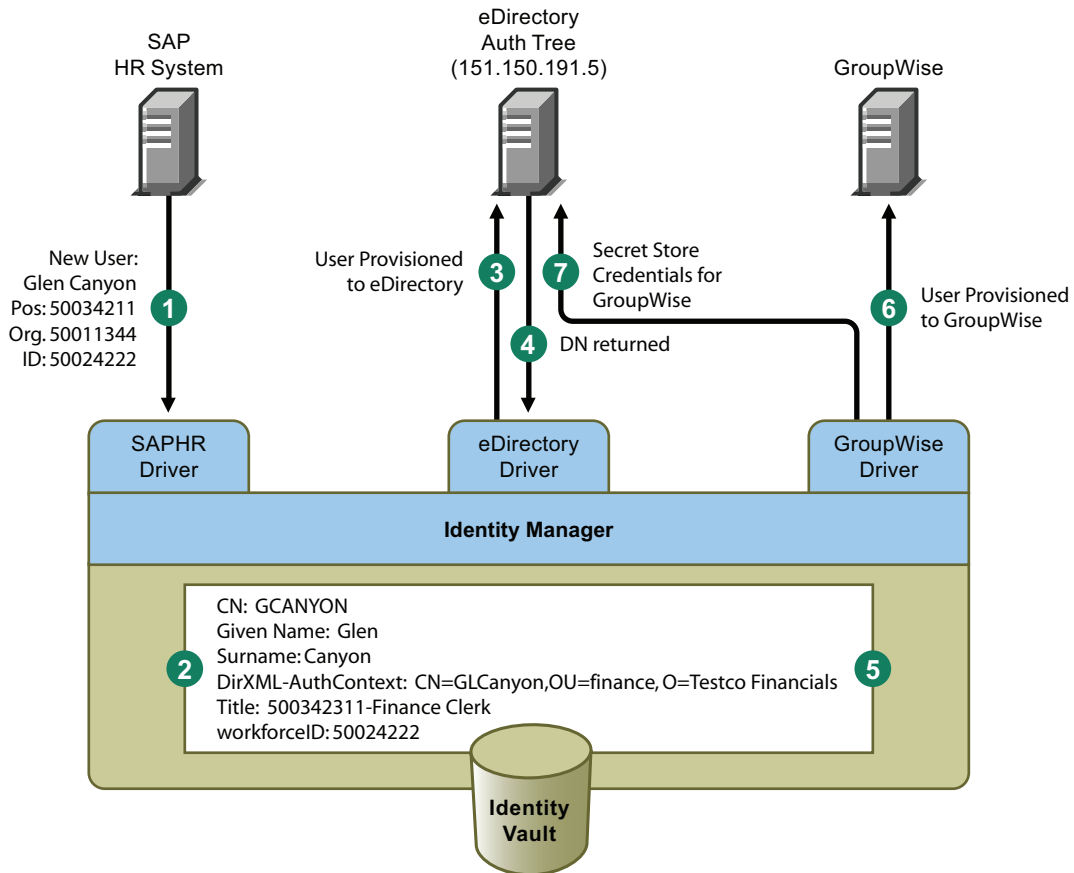


Figure 4-1 illustrates the following provisioning steps:

1. The SAP HR system publishes the data for a newly hired user named Glen Canyon. The Identity Manager SAP HR driver processes this data.

2. A new User object is created in the Identity Vault with a CN value of GCANYON and a workforceID value of 50024222. Because this user is assigned to the Finance organization of his company, he needs to authenticate to the Finance Department eDirectory server. The Identity Manager eDirectory driver that synchronizes that domain now uses the Identity Vault information.

3. Glen is provisioned to the Finance department eDirectory server.

4. The driver is configured to obtain Glen's fully distinguished LDAP name: CN=GLCanyon,OU=finance,O=Testco Financials.

5. The LDAP name is placed into the DirXML-AuthContext (extension of User object, copy of DirXML-ADContext) attribute of the GCANYON user in the Identity Vault.

   Now that the required attributes are available in the Identity Vault, the GroupWise driver processes the attributes of the GCANYON object.

6. Because Glen is in the Finance organization, the driver provisions a GroupWise account for GCANYON on the Finance Departments GroupWise domain server.

7. After the account creation is successful, the GroupWise driver policies provision Glen's GroupWise authentication credentials to the secret store of his eDirectory user account.

When Glen authenticates to his company's Web site from the Internet, an iChain server can use the SecretStore credentials to form-fill his authentication to his secure GroupWise e-mail account, eliminating the need for him to enter his GroupWise credentials and also providing additional security for the company's resources.

# Implementing Novell Credential Provisioning Policies with Novell SecretStore

5

The implementation of Novell® Credential Provisioning policies with Novell SecretStore® is very customizable. The steps to implement it are different depending upon the platforms SecretStore is installed on, the applications that are provisioned, and which Identity Manager drivers are involved.

To implement Credential Provisioning policies with SecretStore:

- Section 5.1, "Requirements for Credential Provisioning Policies with Novell SecretStore," on page 43
- Section 5.2, "Determining Deployment Configuration Parameters for Novell SecretStore," on page 43
- Section 5.3, "Creating a Repository Object for Novell SecretStore," on page 46
- Section 5.4, "Creating an Application Object for Novell SecretStore," on page 54
- Section 5.5, "Creating Credential Provisioning Policies for Novell SecretStore," on page 61
- Section 5.6, "Example Credential Provisioning Policies," on page 63
- Section 5.7, "Operation Data Caching," on page 64
- Section 5.8, "SecretStore Provisioning," on page 65
- Section 5.9, "SecretStore Deprovisioning," on page 65

## 5.1 Requirements for Credential Provisioning Policies with Novell SecretStore

In order to use Credential Provisioning Policies with SecretStore, the following items must be in place:

- Identity Manager 3.0.1 or above
- eDirectory™ 8.7x or eDirectory 8.8.1 or above; eDirectory 8.8 is not supported.
- Verify that `jsso.jar`, `idmcp.jar`, and `jnet.jar` are in the standard location for Identity Manager Java libraries.
- Novell SecretStore 3.3 or above

After you have verified your environment meets the requirements, proceed to Section 5.2, "Determining Deployment Configuration Parameters for Novell SecretStore," on page 43.

## 5.2 Determining Deployment Configuration Parameters for Novell SecretStore

In order to provide the synchronization functionality described in the deployment scenario illustrated in Figure 4-1, the first step is to gather all of the business process information related to

the Identity Manager and SecretStore environments. You can print Table 5-1, "Credential Provisioning Policies Worksheet for SecretStore," on page 44, and use it as a worksheet to record the information.

**Table 5-1**  *Credential Provisioning Policies Worksheet for SecretStore*

| Configuration Information Needed | Information |
| --- | --- |
| 1) Which applications will be configured for Web Single Sign-On provisioning? | |
| 2) The DNS name or IP address of the SecretStore repository server. | |
| 3) The SSL LDAP port for the SecretStore repository server. | |
| 4) The fully qualified LDAP distinguished name of the administrator for the SecretStore repository server. | |
| 5) The password of the administrator for the SecretStore repository server. | |
| 6) The full path and the name of the SSL certificate exported from the SecretStore server. The certificate must be local to the Identity Manager server. | |
| 7) Determine if SecretStore repositories will be used by multiple drivers or if each driver will use a separate repository. | |
| 8) Record the type of SecretStore secret that is being used. There are two supported types of secrets: <br><br>◆ A: Application Secret (SS_App: prefix) <br>◆ C: Credential Set Secret (SS_CredSet: prefix) | |
| 9) The application ID or Credential Set name for each provisioned application. | |
| 10) List all required authentication keys for each application, such as Username and Password. They might be different for each application. | |
| 11) Determine if any of the authentication key values can be set with a static value. | |
| 12) For non-static values that are or can be different for each user, make a note of the source of the non-static information (event information or Identity Vault attribute values.) | |

| Configuration Information Needed | Information |
| --- | --- |
| 13) If you are implementing SecretStore provisioning on a driver that is also synchronizing a password to the target application, determine if the SecretStore provisioning takes place before or after the password is set in the target application server. | |
| 14) The name of the Driver object where the repository and application objects are to be stored. (Can be different drivers.) | |
| 15) Determine the DN of the User objects for the target application. | |

## 5.2.1 Example Provisioning Configuration Data

Using the provisioning scenario in , the following example data provisions a user's SecretStore credentials for the Finance department's GroupWise® domain server onto users in the Finance eDirectory authentication tree:

**Table 5-2**  *Example Credential Provisioning Policies Worksheet for SecretStore*

| Configuration Information Needed | Information |
| --- | --- |
| 1) Which applications will be configured for Web Single Sign-On provisioning? | GroupWise |
| 2) The DNS name or IP address of the SecretStore repository server. | 151.150.191.5 |
| 3) The SSL LDAP port for the SecretStore repository server. | 636 |
| 4) The fully qualified LDAP distinguished name of the administrator for the SecretStore repository server. | cn=admin,ou=finance,o=Tesetco Financials |
| 5) The password of the administrator for the SecretStore repository server. | dixml |
| 6) The full path and the name of the SSL certificate exported from the SecretStore server. The certificate must be local to the Identity Manager server. | `c:\novell\nds\FinanceAD.cer` |
| 7) Determine if SecretStore repositories will be used by multiple drivers or if each driver will use a separate repository. | For this example, there is only one repository. |
| 8) Record the type of SecretStore secret that is being used. There are two supported types of secrets: ◆ A: Application Secret (SS_App: prefix) ◆ C: Credential Set Secret (SS_CredSet: prex) | |

| Configuration Information Needed | Information |
|---|---|
| 9) The application ID or Credential Set name for each provisioned application. | GroupWise_Credentials |
| 10) List all required authentication keys for each application, such as Username and Password. They might be different for each application. | Username<br>Password |
| 11) Determine if any of the authentication key values can be set with a static value. | No static information for this scenario. |
| 12) For non-static values that are or can be different for each user, make a note of the source of the non-static information (event information or Identity Vault attribute values.) | Username: Identity Vault attribute "CN"<br>Password: Event `<password>` |
| 13) If you are implementing SecretStore provisioning on a driver that is also synchronizing a password to the target application, determine if the SecretStore provisioning takes place before or after the password is set in the target application server. | After |
| 14) The name of the Driver object where the repository and application objects are to be stored. (Can be different drivers.) | GroupWise-Finance driver |
| 15) Determine the DN of the User objects for the target application. | Identity Vault attribute "DirXML-ADContext" |

Miscellaneous Environment Information:

- The Finance department eDirectory tree serves as the SecretStore repository for all Finance applications.

- All finance department provisioning drivers are in a driver set called Finance Drivers.

- The GroupWise account must be deleted and the SecretStore credentials for the GroupWise user account must be removed from the eDirectory user when the Identity Vault attribute employeeStatus is set to the value "I".

As can be seen from the data gathered, the SecretStore repository information is global for all drivers that provision Finance department applications. In addition, all provisioning information can be statically configured, with the exception of the GroupWise login parameters Username, Password, and Target User DN.

After all of the configuration data has been determined, proceed to .

# 5.3  Creating a Repository Object for Novell SecretStore

Repository objects store static configuration information for SecretStore. Repository information is independent from the applications that consume the application credentials. This information is applicable for all provisioning events regardless of the connected system (for example SAP, PeopleSoft, Notes, etc.) The repository object can be created in Designer or iManager.
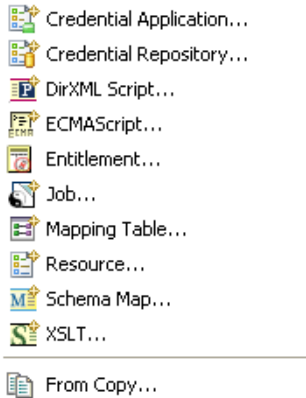
-

## 5.3.1 Creating a Repository Object for Novell SecretStore in Designer

The following is one of many methods you can use to create the repository object in Designer.
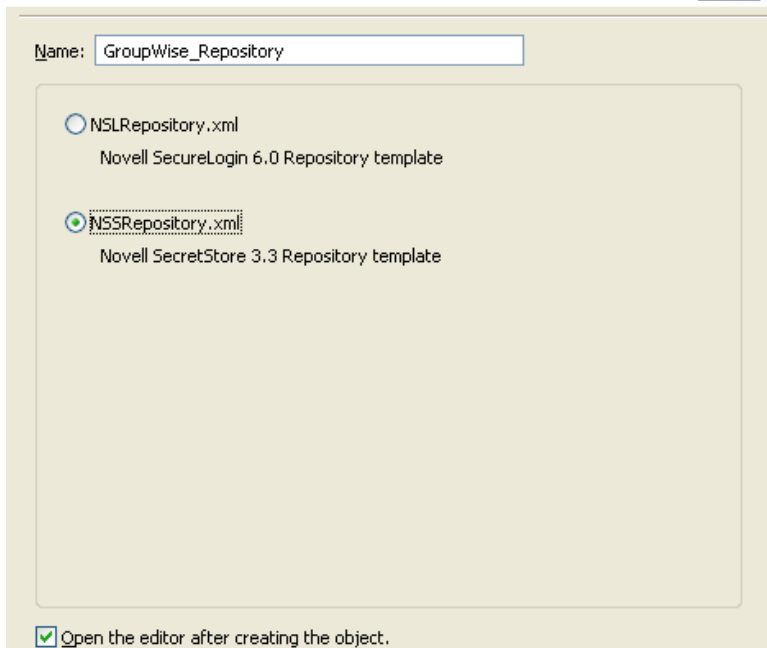
**1** In the outline view, right-click the driver object where you want to store the repository object.

**2** Click *New > Credential Repository*.



**3** Specify a name for the repository object.

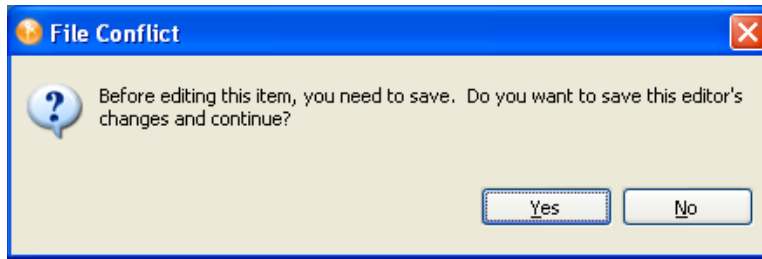**4** Select *NSSRepository.xml* to use the SecretStore template.



Verify that the *Open the editor after creating the object* check box is selected.

**5** Click *OK*.

**6** Click *Yes* to save the new repository object.



**7** Specify the DNS name or IP address of the SecretStore server. (See worksheet item 2).

SecretStore Server Name or Address:

151.150.191.5

**8** Specify the SSL port for the SecretStore server. (See worksheet item 3).

SecretStore Server SSL Port: 636  ⓘ

**9** Specify the full path to the SSL certificate exported from the SecretStore server. The path must include the certificate name and must be local to the Identity Manager server. (See worksheet item 6).

SecretStore Server SSL Certificate Path:

c:\novell\nds\FinaceSrv.der

Refer to the Certificate Server (http://www.novell.com/documentation/crt32/index.html) documentation for the information on how to export the SSL certificate.

**10** Specify the fully qualified LDAP distinguished name of the SecretStore administrator. (See worksheet item 4).

Use Enhanced Protection Flag: Disabled ▾  ⓘ

**11** Click *Set password*.

SecretStore Administrator Password:  [Set Password...] [Remove Password]

**12** Specify the SecretStore administrator's password twice, then click OK. (See worksheet item 5).



**13** Review the information, then click the *Save* icon 💾 to save the information.

**14** (Optional) If you want to create other configuration parameters for the repository object, click the *Add new item* icon ✚ .



**14a** Specify a name for the parameter.

**14b** Specify a display name for the parameter.

**14c** Specify a description of the parameter for your reference.

The parameter is stored as a string.



**14d** Click *OK*.

**14e** Click the *Save* icon 💾 to save the repository object.

After the repository object is created, proceed to .

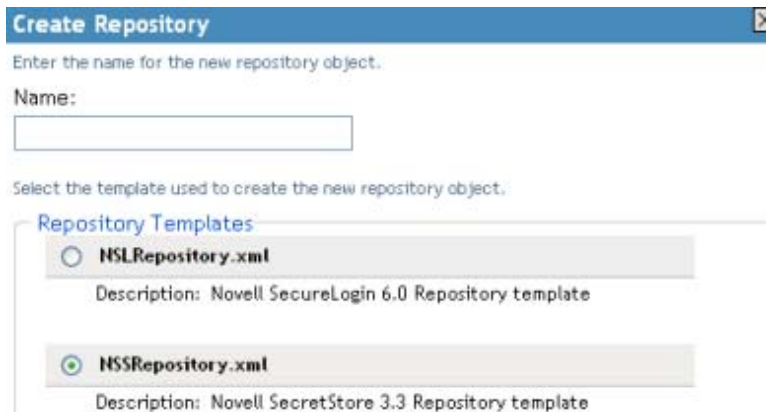## 5.3.2  Creating a Repository Object for Novell SecretStore in iManager

1  In iManager, select *Credential Provisioning > Configuration*.

2  Browse to and select the Driver object where the repository object will be stored.

**3** Click *New* to create a repository.

Repositories    Applications

New...   |   Delete

Name

No repositories were found- Select 'New'

**4** Specify a name for the repository object.

**5** Select *NSSRepository.xml* to use the SecretStore template to create a repository.

**Create Repository**      ✕

Enter the name for the new repository object.

Name:

Select the template used to create the new repository object.

Repository Templates

○ **NSLRepository.xml**

    Description: Novell SecureLogin 6.0 Repository template

◉ **NSSRepository.xml**

    Description: Novell SecretStore 3.3 Repository template

**6** Click *OK*.

**7** Specify the DNS name or IP address of the SecretStore server. (See worksheet item 2).

☐   SecretStore Server Name or Address ⓘ

**8** Specify the SSL port for the SecretStore server. (See worksheet item 3).

☐   SecretStore Server SSL Port ⓘ

**9** Specify the full path to the SSL certificate exported from the SecretStore server. The path must include the certificate name and must be local to the Identity Manager server. (See worksheet item 6).

☐   SecretStore Server SSL Certificate Path ⓘ

Refer to the Certificate Server (http://www.novell.com/documentation/crt32/index.html) documentation for the information on how to export the SSL certificate.

**10** Specify the fully qualified LDAP distinguished name of the SecretStore administrator. (See worksheet item 4).

SecretStore Administrator ⓘ

**11** Click *Set password*.

SecretStore Administrator Password ⓘ    Set password

**12** Specify the SecretStore administrator's password twice, then click *OK*. (See worksheet item 5).

**Enter Password** ☒

Enter password:

Reenter password:

OK    Cancel

**13** Review the values specified, then click *OK*.

**14** (Optional) If you want to create other configuration parameters for the repository object, click *New*.
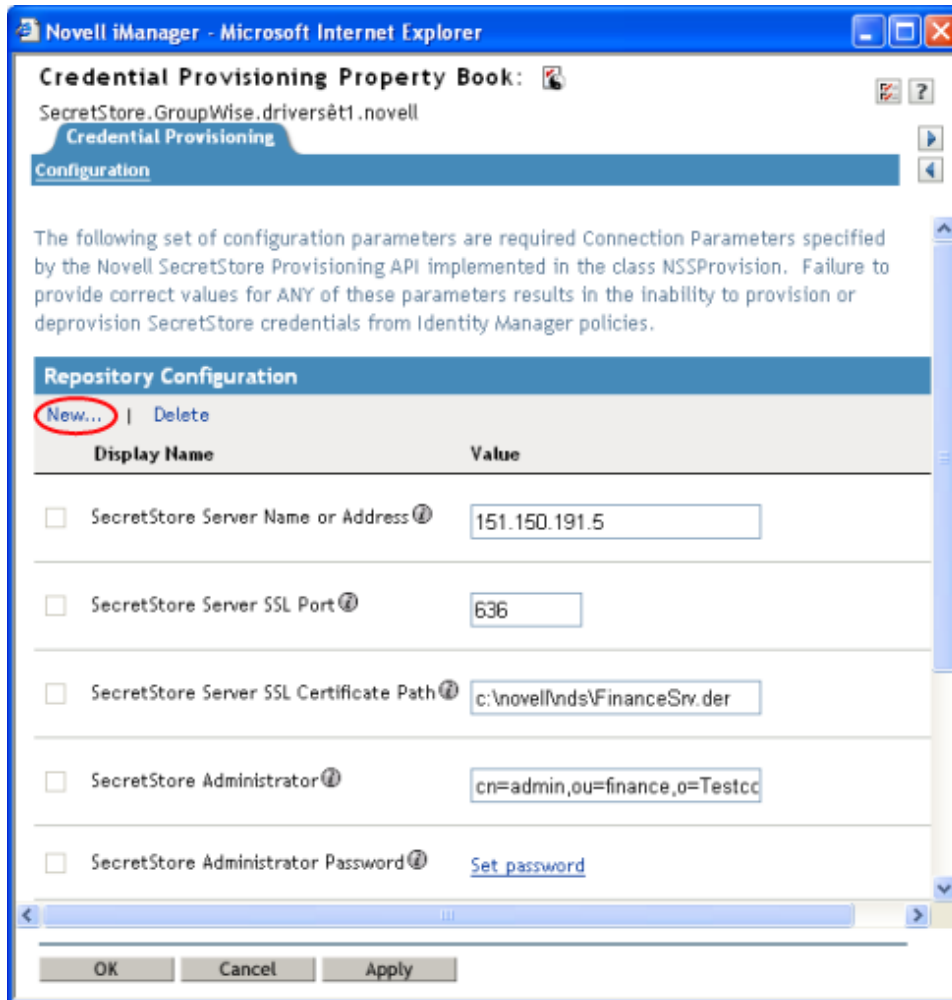


The example information is from the scenario in .

**14a** Specify a name for the parameter.

**14b** Specify a display name for the parameter.

**14c** Specify a description of the parameter for your reference.

The parameter is stored as a string.

**Global Configuration Value Definition**

Global Configuration Values are a means through which the behavior of an Identity Manager driver configuration can be changed without requiring any policy to be changed.

Name:

Display name:

Description:

Type:
string

**14d** Click *OK*.

After the repository object is created, proceed to .

# 5.4 Creating an Application Object for Novell SecretStore

Applications store static configuration parameter values for SecretStore. Application information is specific to the applications that are consuming the application credential (for example, GroupWise client information or SAP database client information). The application objects can be created in Designer or iManager.

## 5.4.1 Creating an Application Object for Novell SecretStore in Designer

The following is one of many methods you can use to create the application in Designer.

**1** In the Outline view, right-click the driver object where you want to store the application object.

**2** Click *New > Credential Application*.



**3** Specify a name for the application object.

**4** Select *NSSApplication.xml* to use the SecretStore template.



Verify that the *Open the editor after creating the object* check box is selected.

**5** Click *OK*.

**6** Click *Yes* to save the new application object.



**7** Specify the SecretStore Application ID. (See worksheet item 9).
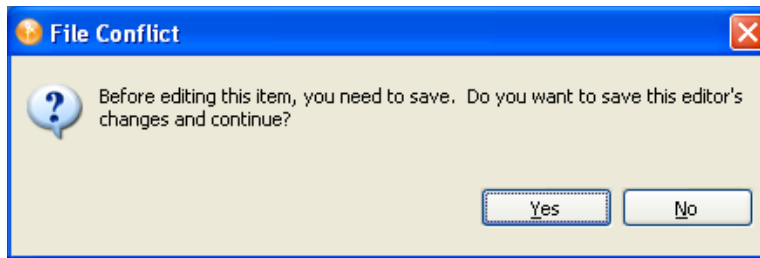
SecretStore Application ID:

GroupWise_Credentials

**8** Select the *SecretStore Secret Type*. (See worksheet item 8).

SecretStore Secret Type: Shared ⓘ

**9** Select the *SecretStore Shared Secret Type*. (See worksheet item 8).

SecretStore Shared Secret Type: Credential Set ⓘ

**10** Select whether the *SecretStore Use Enhanced Protection Flag* is *Disabled* or *Enabled*.

Use Enhanced Protection Flag: Disabled ⓘ

**11** Click *Set Password* to set the *Enhanced Protection Password* if it is enabled.

Enhanced Protection Password: Set Password... Remove Password ⓘ

**12** Specify the password twice, then click *OK*.

**13** Click the *Save* icon 🖫 to save the application.

**14** Click the *Add new item* icon ✚ to add the authentication keys required for the application.
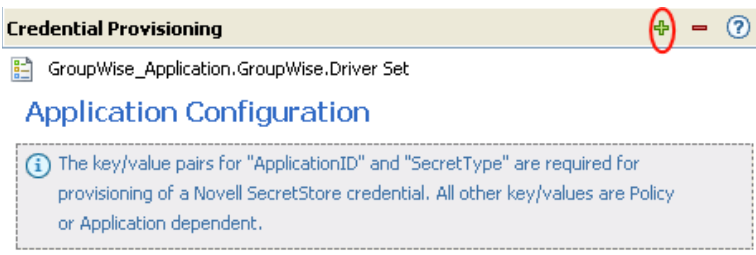


**14a** Specify a name for the authentication key.

**14b** Specify a display name for the authentication key.

**14c** Specify a description of the authentication key for your reference.

The authentication key is stored as a string.



**14d** Click *OK*.

**14e** Repeat Step 14 for each new authentication key that needs to be entered.

**15** Specify the authentication key value, if it is a static value that is shared by all user credentials.

**16** Click the *Save* icon 🖫 to save the application.

After the application object is created, proceed to Section 5.5, "Creating Credential Provisioning Policies for Novell SecretStore," on page 61.

## 5.4.2  Creating an Application Object for Novell SecretStore in iManager

**1** In iManager, select *Credential Provisioning > Configuration*.

**2** Browse to and select the Driver object where the application object will be stored, then click *OK*.
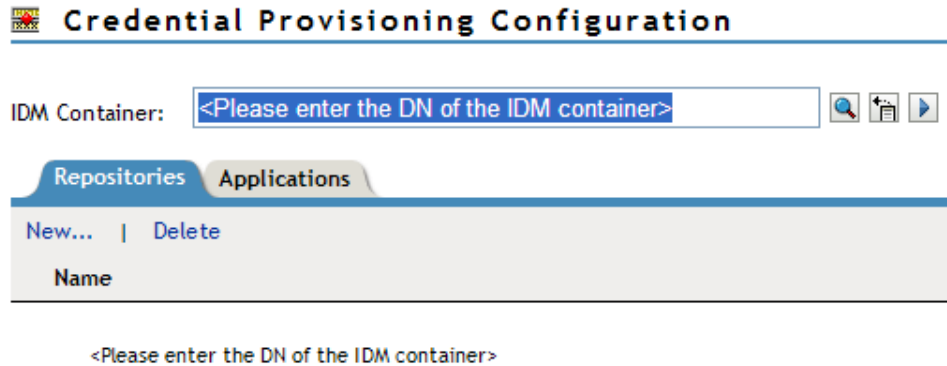


**3** Select the *Applications* tab, then click *New*.
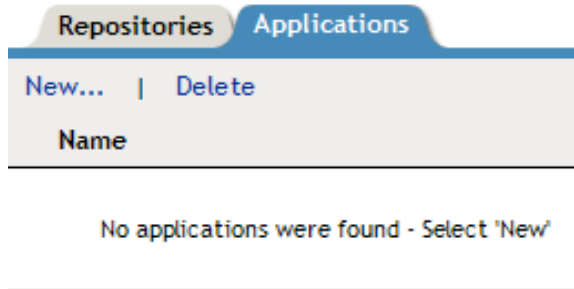


**4** Specify a name for the application object

**5** Select *NSSApplication.xml* to use the SecretStore template to create an application.
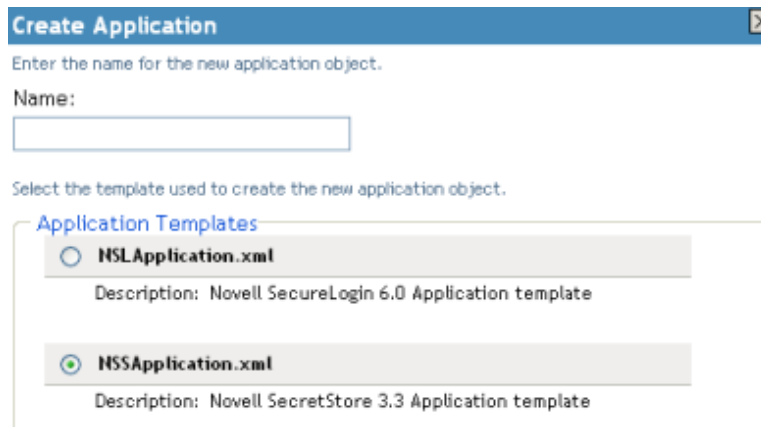


**6** Click *OK*.

**7** Specify the *SecretStore Application ID*. (See worksheet item 9).

☐ SecretStore Application IDⓘ [                    ]

**8** Select the *SecretStore Secret Type*. (See worksheet item 7). The SecretStore type is *Shared* or *Not Shared*.

☐ SecretStore Secret Typeⓘ [ Shared ▾ ]

**9** Select the *SecretStore Shared Secret Type*. (See worksheet item 8). The Shared SecretStore type is *Credential Set* or *Application*.

☐ SecretStore Shared Secret Typeⓘ [ Credential Set ▾ ]

**10** Select whether the SecretStore *Use Enhanced Protection Flag* is *Disabled* or *Enabled*.

☐ Use Enhanced Protection Flagⓘ [ Disabled ▾ ]

**11** Click *Set password* to set the *Enhanced Protection Password* if it is enabled.

☐ Enhanced Protection Passwordⓘ   Set password

**12** Specify the password twice, then click *OK*.

**Enter Password** ☒

Enter password:
[                    ]

Reenter password:
[                    ]

[ OK ]   [ Cancel ]

**13** Click *New* to create an authentication key that the application requires. (See worksheet item 10).

**13a** Specify a name for the authentication key.

**13b** Specify a display name for the authentication key.

**13c** Specify a description of the authentication key for your reference.

The authentication key is stored as a string.



**13d** Click *OK*.

**13e** Repeat Step 13 for each authentication key that the application requires.

**14** Specify the value of the authentication key, if it is static, then click *OK*.
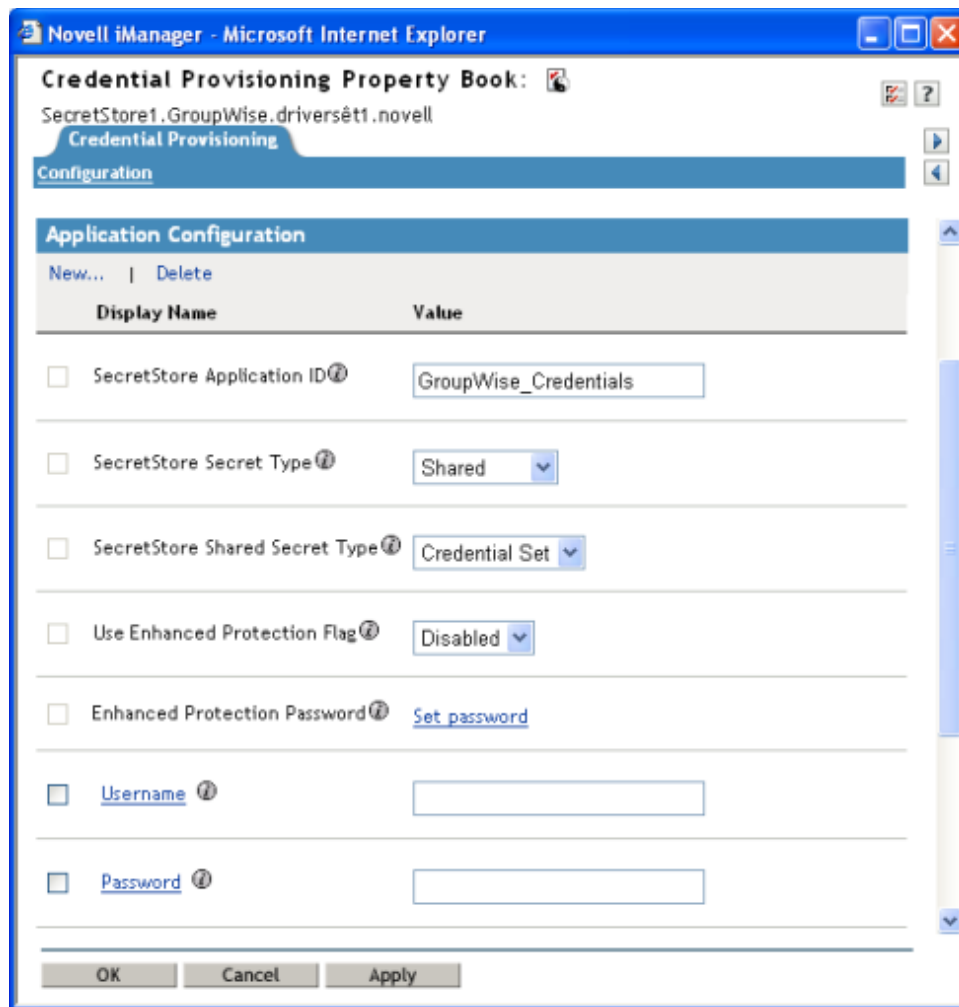


After the application object is created, proceed to .

# 5.5 Creating Credential Provisioning Policies for Novell SecretStore

After the repository and application objects are created, policies need to be created to provision SecretStore information. The policies can be created in Designer or iManager.

## 5.5.1 Creating Credential Provisioning Policies for Novell SecretStore in Designer

The policies use the information stored in the repository and application objects.

**1** In the Policy Builder, create a new policy.

**2** (Optional) To clear the SSO credential, so objects can be deprovisioned, select the *clear SSO credential* action, then fill in the following fields:

Do  clear SSO credential  ▾  ⑦

Specify credential repository object DN: *  [                    ] 🔍

☑ Render browsed DN relative to policy

Specify target user DN: *  [                    ] ▦

Populate the following from an application object

Specify application credential ID: *  [                    ]

Specify login parameter strings:  [                    ] ▦

- ◆ **Specify Credential Repository Object DN:** Browse to and select the repository object. (See worksheet item 8).

- ◆ **Specify Target User DN:** Create the DN of the target users by using the Argument Builder. (See worksheet item 15).

- ◆ **Specify Application Credential ID:** Specify the application ID. (See worksheet item 9).

- ◆ **Specify Login Parameter Strings:** Launch the String Builder and enter each authentication key for the application. (See worksheet item 10).

**3** (Optional) To set the SSO credential when a user object is created or when a password is modified, select the *set SSO credential* action, then fill in the following fields:

Do  set SSO credential  ▾  ⑦

Specify credential repository object DN: *  [                    ] 🔍

☑ Render browsed DN relative to policy

Specify target user DN: *  [                    ] ▦

Populate the following from an application object

Specify application credential ID: *  [                    ]

Specify login parameter strings:  [                    ] ▦

- ◆ **Specify Credential Repository Object DN:** Browse to and select the repository object. (See worksheet item 8).

- ◆ **Specify Target User DN:** Create the DN of the target users by using the Argument Builder. (See worksheet item 15).

- ◆ **Specify Application Credential ID:** Specify the application ID. (See worksheet item 9).

- ◆ **Specify Login Parameter Strings:** Launch the String Builder and enter each authentication key for the application. (See worksheet item 10).

## 5.5.2  Configuring Credential Provisioning Policies for Novell SecretStore in iManager

The policies use the information stored in the repository and application objects.

**1** In the Policy Builder, create a new policy.

**2** (Optional) To clear the SSO credential, so objects can be deprovisioned, select the *clear SSO credential* action, then fill in the following fields:



- ◆ **Enter Credential Repository Object DN:** Browse to and select the repository object. (See worksheet item 8).

- ◆ **Enter Target User DN:** Create the DN of the target users by using the Argument Builder. (See worksheet item 15).

- ◆ **Enter Application Credential ID:** Specify the application ID. (See worksheet item 9).

- ◆ **Enter Login Parameter Strings:** Launch the String Builder and enter each authentication key for the application. (See worksheet item 10).

**3** (Optional) To set the SSO credential when a user object is created or when a password is modified, select the *set SSO credential* action, then fill in the following fields:



- ◆ **Enter Credential Repository Object DN:** Browse to and select the repository object. (See worksheet item 8).

- ◆ **Enter Target User DN:** Create the DN of the target users by using the Argument Builder. (See worksheet item 15).

- ◆ **Enter Application Credential ID:** Specify the application ID. (See worksheet item 9).

- ◆ **Enter Login Parameter Strings:** Launch the String Builder and enter each authentication key for the application. (See worksheet item 10).

# 5.6 Example Credential Provisioning Policies

The credential provisioning policies can be implemented and customized to meet the needs of your environment. The following example explains how to implement the polices for the scenario presented in Figure 4-1 on page 40.

In the Finance scenario, SecretStore provisioning occurs after a password is successfully set in GroupWise. Most of the necessary parameters are statically configured and available to all policies

through the repository and application objects. However, there are non-static data parameters (CN, password, and DirXML-ADContext) that are available only after the GroupWise user `<add>` or `<modify-password>` commands complete and the `<output>` document is returned from the GroupWise driver shim. The `<output>` document no longer contains any of the Subscriber operation attributes and the User context of the command is lost, thus preventing queries on the object. It is therefore necessary to do the following:

- Make sure the GroupWise driver's Subscriber Create policy enforces the presence of the non-static data parameters.
- Cache the non-static parameters required for the provisioning operation prior to issuing the Subscriber command to the GroupWise driver shim.
- Retrieve cached data for use in SecretStore provisioning after the command completes successfully.

---

**NOTE:** Sample policies are available in XML format on the Identity Manager 3.0 Support Pack 1 media. The filenames are `SampleInputTransform.xml`, `SampleSubCommandTransform.xml`, and `SampleSubEventTransform.xml`. The files are found in the following directories:

- `linux\setup\utilities\cred_prov`
- `nt\dirxml\utilities\cred_prov`
- `nw\dirxml\utilities\cred_prov`

The files are installed to the Identity Manager server, if Credential Provisioning Sample Policies is selected during the installation of the utilities. The sample policies are installed to the following locations, depending upon the platform:

- Windows: `C:\Novell\NDS\DirXMLUtilities` (default; the user can change it during install)
- NetWare®: `SYS:\System\DirXmlUtilities`
- Linux (eDir 8.7): `/usr/lib/dirxml/rules/credprov`
- Linux (eDir 8.8.1): `/opt/novell/eDirectory/lib/dirxml/rules/credprov` (default; the user can change it during install)

The sample policies provide a starting point to develop a policy that works for your environment.

---

# 5.7  Operation Data Caching

The mechanism that is available for required operation data caching required is the `<operation-data>` element. Because you might need to provision the SecretStore account from either an `<add>` or `<modify-password>` command, a logical place to implement the non-static data caching policy is in the Subscriber Command Transformation policy. The following example shows a typical SecretStore Provisioning element:

```
<operation-data>
    <nss-sync-data>
        <nss-target-user-dn>
cn=GLCANYON,ou=finance,o=Testco Financials
        </nss-target-user-dn>
        <nss-app-username>GCANYON</nss-app-username>
```

```
        <password><!-- content suppressed --></password>
        <nss-passphrase-answer>50024222</nss-passphrase-answer>
    </nss-sync-data>
</operation-data>
```

In the sample Finance department scenario from , the following values are needed to populate the operation data payload:

- The `<nss-target-user-dn>` element is populated with the value of the DirXML-ADContext attribute from the Identity Vault, which was set by the eDirectory driver. To ensure that the GroupWise driver is notified when the value is set by the eDirectory driver, make sure you add DirXML-ADContext to the Subscriber filter as a notify attribute.

- The `<nss-app-username>` element is populated by the value of the CN attribute in the Identity Vault.

- The password element is populated with the value of the `<password>` element in the `<add>` or `<modify-password>` command.

## 5.8  SecretStore Provisioning

In the sample scenario, the first available location from which the operation data can be retrieved and utilized for SecretStore credential provisioning is in the driver's Input Transformation policy. In the sample scenario, two policies are implemented:

- Set SecretStore Credentials after successful password synchronization
- Remove SecretStore Credentials if Application User Deleted (Identity Vault object not deleted)

  **NOTE:** There is a sample policy in the `SampleInputTransform.xml` file that sets the SecretStore credentials after a successful password synchronization occurs. The file is located in the `cred_prov` folder in the utilities directory on the Identity Manager 3.0 Support Pack 1 media.

The Set SecretStore Credentials policy needs to make sure the provisioning happens only if the returned command status is Success and the previously set `<operation-data>` is present.

## 5.9  SecretStore Deprovisioning

There are many scenarios that can utilize a policy in which a user account for a connected application is deleted and the Identity Vault account remains. In the Finance scenario, there is a requirement to delete the GroupWise account and deprovision the SecretStore credentials when the user's Identity Vault employeeStatus attribute value is set to "I". To handle this situation, the GroupWise driver's Subscriber Event Transformation contains a policy to transform the modify attribute value into an object delete. Because the eDirectory account name is still needed after the `<delete>` command is completed, the `<operation-data>` event needs to be set on the `<delete>` command so it is available to the SecretStore deprovisioning policy in the Input Transformation policy.

```
<operation-data>
    <nss-sync-data>
        <nss-target-user-dn>cn=GLCANYON,ou=finance,o=Testco
Financials
        </nss-targer-user-dn>
```

```
      </nss-sync-data>
</operation-data>
```

The policy for transforming the `<modify>` event into a `<delete>` and creating this element is available in XML format in a file called `SampleSubEventTransform.xml` files in the `cred_prov` folder in the `utilities` directory on the Identity Manager 3.0 Support Pack 1 media.

After the policy is created, proceed to .

# Managing Novell Credential Provisioning Policies
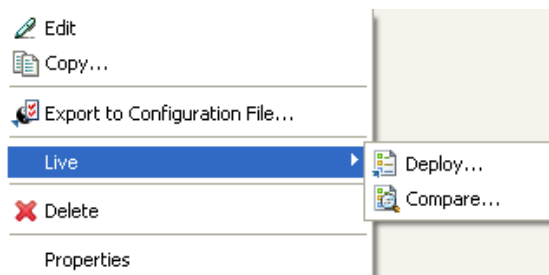
There are additional tasks you can perform to manage the Novell® Credential Provisioning policies after they are implemented.

## 6.1 Managing the Repository and Application Objects

To manage repository and application objects (resource objects):

**1** In the Outline view, right-click the resource object.

**2** Select the desired task.



- ◆ **Edit:** Lets you edit the resource object.
- ◆ **Copy:** Copies the resource object.
- ◆ **Export to Configuration File:** Saves the resource object as an XML file. Use this option to create a backup of the resource object.
- ◆ **Live > Deploy:** Deploys the resource object into the Identity Vault.
- ◆ **Live > Compare:** Compares the resource object to the corresponding object in the Identity Vault.
- ◆ **Delete:** Deletes the resource object.
- ◆ **Properties:** Lets you rename the resource object.

## 6.2 Managing the Credential Provisioning Policies

**1** In the Outline view, right-click the policy.

**2** Select the desired task.
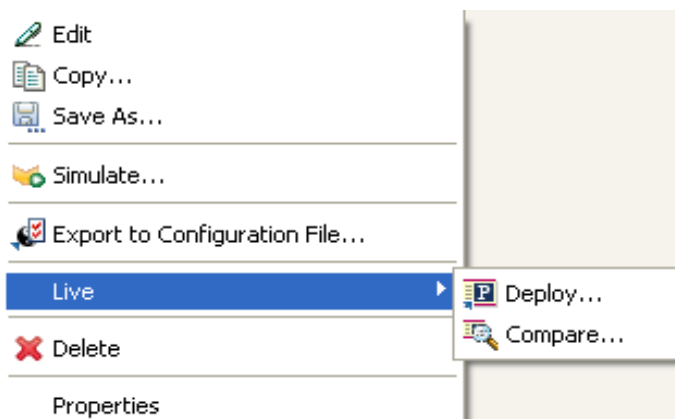


- **Edit:** Lets you edit the policy.
- **Copy:** Copies the policy.
- **Save As:** Saves a copy of the policy with another name.
- **Simulate:** Lets you test the policy in Designer before it is deployed into the Identity Vault.
- **Export to Configuration File:** Saves the policy as an XML file. Use this option to create a backup of the policy.
- **Live > Deploy:** Deploys the policy into the Identity Vault.
- **Live > Compare:** Compares the policy to the corresponding policy in the Identity Vault.
- **Delete:** Deletes the policy.
- **Properties:** Lets you rename the policy.