

OpenText™ Identity Manager CE

24.4 (v4.10)

Credential Provisioning

December 2024

Legal Notice

Copyright 2009 - 2024 Open Text.

The only warranties for products and services of Open Text and its affiliates and licensors ("Open Text") are as may be set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Open Text shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

For additional information, such as certification-related notices and trademarks, see <http://www.microfocus.com/about/legal/> (<http://www.microfocus.com/about/legal/>).

Contents

About this Book and the Library	5
Part I Implementing Credential Provisioning with OpenText SecureLogin	7
1 Overview	9
2 Requirements	13
3 Implementing Credential Provisioning	15
Extending the LDAP Schema	15
Determining Deployment Configuration Parameters for NetIQ SecureLogin	16
Example Provisioning Configuration Data	17
Creating a Repository Object	19
Creating an Application Object	21
Creating Credential Provisioning Policies	22
Example Credential Provisioning Policies	24
Operation Data Caching	25
SecureLogin Provisioning	25
SecureLogin Deprovisioning	26
4 Managing Credential Provisioning	27
Managing the Repository and Application Objects	27
Managing the Credential Provisioning Policies	27

About this Book and the Library

Credential Provisioning for OpenText Identity Manager enhances the user provisioning abilities of any OpenText Identity Manager driver by providing the capability to provision application credentials to the OpenText SecureLogin credential repository. Additionally, you can provision the 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 OpenText 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 can use Identity Manager policies to automatically de-provision application credentials to prevent access to application data.

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

Intended Audience

This guide is intended for OpenText Identity Manager administrators.

Other Information in the Library

For more information about the library for OpenText Identity Manager, see the [OpenText Identity Manager documentation website \(https://www.netiq.com/documentation/identity-manager-49/\)](https://www.netiq.com/documentation/identity-manager-49/).

Implementing Credential Provisioning with OpenText SecureLogin

The following sections provide the concepts and instructions required to implement credential provisioning with OpenText SecureLogin:

- ♦ [Chapter 1, “Overview,” on page 9](#)
- ♦ [Chapter 2, “Requirements,” on page 13](#)
- ♦ [Chapter 3, “Implementing Credential Provisioning,” on page 15](#)
- ♦ [Chapter 4, “Managing Credential Provisioning,” on page 27](#)

1 Overview

Credential Provisioning allows you to automatically provision application credentials that OpenText SecureLogin supports. This section documents the steps required to configure objects and policies in OpenText Identity Manager. It does not contain deployment and configuration information for any OpenText SecureLogin components. For documentation, see the [OpenText SecureLogin 6.1 Documentation Web Site \(https://www.netiq.com/documentation/securelogin-85/\)](https://www.netiq.com/documentation/securelogin-85/).

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 Credential Provisioning,” 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.
- OpenText 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 OpenText 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 “[Password Generation](#)” in the *OpenText™ Identity Manager Security Guide* for using random password generation.

[Figure 1-1, “Credential Provisioning with SecureLogin,” on page 10](#) shows a typical, yet simple, scenario involving the provisioning of the OpenText 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 OpenText 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 OpenText SecureLogin credential repository. As users are subsequently provisioned by OpenText Identity Manager to the various finance applications, their credentials for those systems are synchronized to the SecureLogin store in Active Directory.

[Figure 1-1](#) 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 1-1 Credential Provisioning with SecureLogin

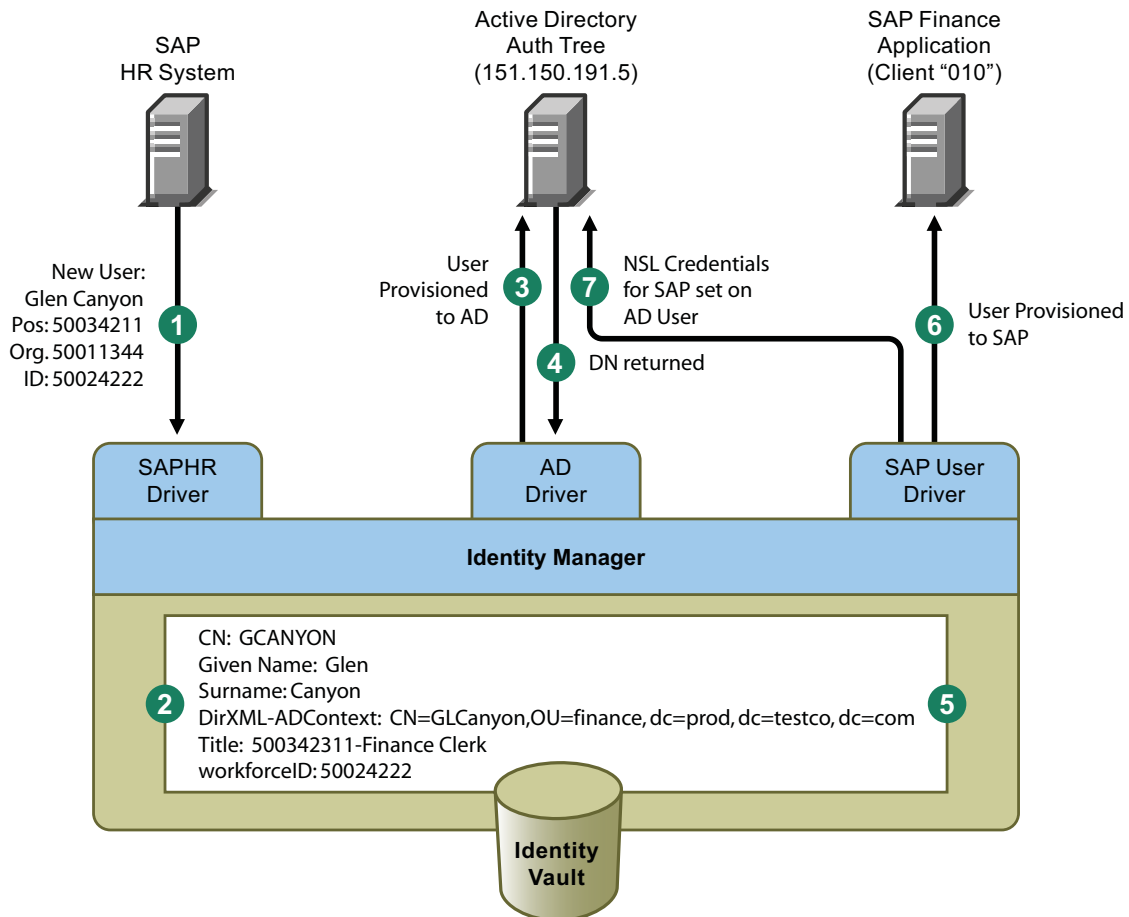


Figure 1-1 illustrates the following steps:

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=finance,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.

2 Requirements

In order to use credential provisioning with OpenText SecureLogin, the following must be in place:

- ♦ Identity Manager 3.6 or above
- ♦ `jsso.jar`, `idmcp.jar`, and `jnet.jar` must reside in the standard location for OpenText Identity Manager Java libraries.
- ♦ OpenText SecureLogin 6.0 or above

3 Implementing Credential Provisioning

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

- ♦ “Extending the LDAP Schema” on page 15
- ♦ “Determining Deployment Configuration Parameters for NetIQ SecureLogin” on page 16
- ♦ “Creating a Repository Object” on page 19
- ♦ “Creating an Application Object” on page 21
- ♦ “Creating Credential Provisioning Policies” on page 22
- ♦ “Example Credential Provisioning Policies” on page 24
- ♦ “Operation Data Caching” on page 25
- ♦ “SecureLogin Provisioning” on page 25
- ♦ “SecureLogin Deprovisioning” on page 26

Extending the LDAP Schema

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 OpenText eDirectory and are required in order for the OpenText SecureLogin product to function. The provisioning API provided in Identity Manager 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.

If these two tools have not been run, see “Installing” (https://www.netiq.com/documentation/securelogin-85/installation_guide/data/front.html) in the *OpenText SecureLogin 6.1 Installation Guide*.

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

- 1 In Identity Console, click **LDAP Configuration**.
- 2 Specify the name, type or context of the LDAP Group object associated with your eDirectory servers that host SecureLogin.
- 3 Select the LDAP Group object from the search list.
- 4 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

- 5 After the schema is extended, proceed to [“Determining Deployment Configuration Parameters for NetIQ SecureLogin” on page 16](#).

Determining Deployment Configuration Parameters for NetIQ SecureLogin

In order to provide the synchronization functionality described in the deployment scenario illustrated in [Figure 1-1, “Credential Provisioning with SecureLogin,” on page 10](#), the first step is to gather all of the business process information related to the OpenText Identity Manager and OpenText SecureLogin environments. You can print the following table 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.	

Configuration Information Needed	Information
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).	
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:

Example Provisioning Configuration Data

Using the provisioning scenario in [Figure 1-1, “Credential Provisioning with SecureLogin,” on page 10](#), 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
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 SAP Client 010 Login Parameter Language SAP Client 010 Login Parameter Username 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" 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" 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

Configuration Information Needed	Information
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?" 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 ["Creating a Repository Object" on page 19](#).

Creating a Repository Object



Repository objects store static configuration information for OpenText 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.

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** .
- 3 Specify a name for the repository object.
- 4 Select **NSLRepository.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**, in the file conflict window, to save the new repository object.
- 7 Use the following information to complete the creation of the Repository object.

Field	Description
SecureLogin Server Name or Address	Specify the DNS name or IP address of the Secure Login server. (See worksheet item 3).
SecureLogin Server SSL Port	Specify the SSL port for the SecureLogin server. (See worksheet item 4).

Field	Description
SecureLogin Server SSL Certificate Path	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 Administrator	Specify the fully qualified LDAP distinguished name of the SecureLogin administration. (See worksheet item 5).
SecureLogin Administrator Password	Specify the SecureLogin administrator's password twice, then click OK to save the password. (See worksheet item 6).


- 8 Review the information, then click the **Save** icon  to save the information.
- 9 (Optional) If you want to create other configuration parameters for the repository object, click the **Add new item**  icon.
 - 9a Specify a name for the parameter.
 - 9b Specify a display name for the parameter.
 - 9c Specify a description of the parameter for your reference.
The parameter is stored as a string.


Name:

Display name:

Description:

Type:

string 


- 9d Click **OK**.
- 9e Click the **Save** icon  to save the repository object.

After the repository object is created, proceed to [“Creating an Application Object” on page 21](#).

Creating an Application Object

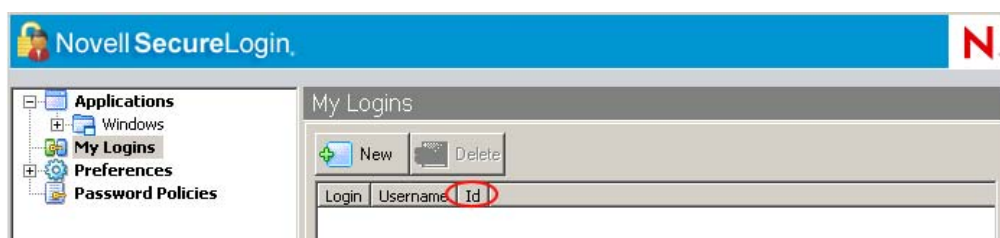
Application objects store application authentication parameter values for OpenText SecureLogin. Application information is specific to the applications that are consuming the application credential (for example, GroupWise client information or SAP database client information). The application objects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



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

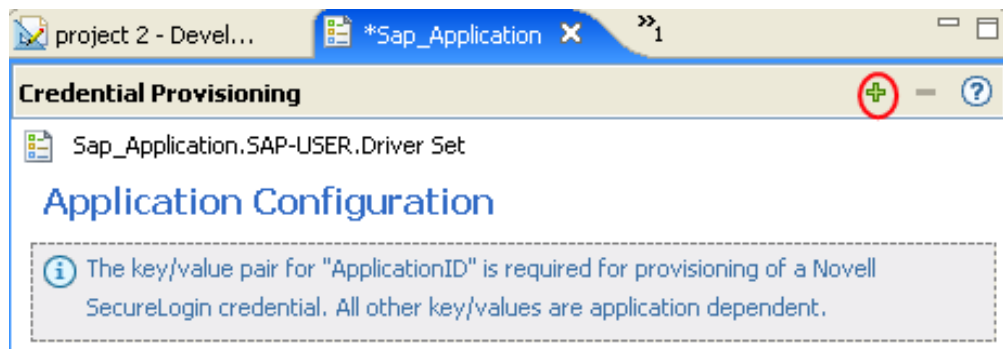
- 1 In the Outline view, right-click the driver object where you want to store the application object.
- 2 Click **New > Credential Application** .
- 3 Specify a name for the application object.
- 4 Select **NSLApplication.xml** to use the SecureLogin template.
Verify that the **Open the editor after creating the object** check box is selected.
- 5 Click **OK**.
- 6 Click **Yes**, in the file conflict window, to save the new application object.
- 7 Specify the SecureLogin Application ID. (See worksheet item 9).

SecureLogin Application ID:

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



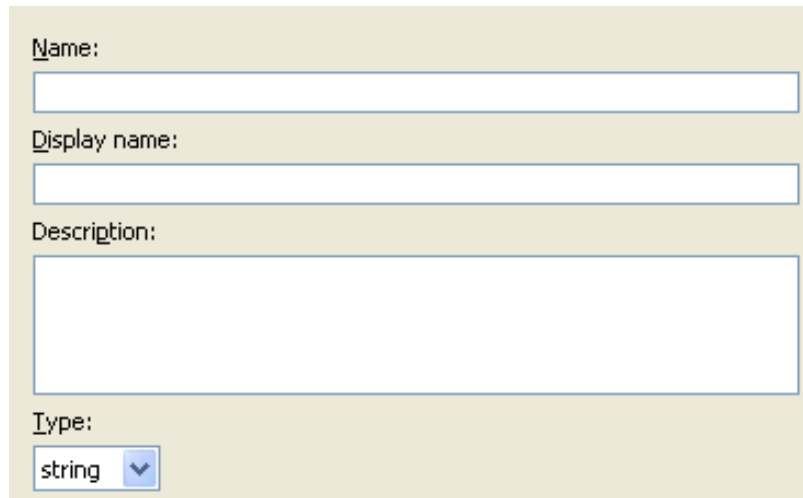
- 8 Click the **Save** icon  to save the application.
- 9 Click the **Add new item** icon  to add the authentication keys required for the application.



- 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.



A form with a light beige background. It contains four labeled input fields: 'Name:' with a single-line text box, 'Display name:' with a single-line text box, 'Description:' with a multi-line text box, and 'Type:' with a dropdown menu. The dropdown menu is currently set to 'string' and has a small blue arrow icon to its right.

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 [“Creating Credential Provisioning Policies” on page 22](#).


Creating Credential Provisioning Policies

After the repository and application objects are created, policies need to be created to provision SecureLogin information. The policies can be created 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 ?


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 ?


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:


Do 

Specify credential repository object DN: * 

☒ Render browsed DN relative to policy

Specify target user DN: * 

Question string: * 

Answer string: * 

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).

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 policies for the scenario presented in [Figure 1-1, “Credential Provisioning with SecureLogin,” on page 10](#).

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 <output> 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.

Sample policies are available in XML format on the Identity Manager media. The filenames are SampleInputTransform.xml, SampleSubCommandTransform.xml, and SampleSubEventTransform.xml. The files are found in the following directory, where platform is linux, windows, aix, or solaris:

```
platform\setup\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
- ♦ Linux: /opt/novell/eDirectory/lib/dirxml/rules/credprov

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

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> <ns1-sync-data> <ns1-target-user-dn>
cn=GLCANYON,ou=finance,dc=prod,dc=testco,dc=com </ns1-target-user-dn>
<ns1-app-username>GCANYON</ns1-app-username> <password><!-- content
suppressed --></password> <ns1-passphrase-answer>50024222</ns1-passphrase-
answer> </ns1-sync-data> </operation-data>
```

In the sample Finance department scenario from [Figure 1-1, “Credential Provisioning with SecureLogin,” on page 10](#), the following values are needed to populate the operation data payload:

- ♦ The `<ns1-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 `<ns1-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 `<ns1-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.

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)

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 [cred_prov folder](#) on the Identity Manager 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.

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> <ns1-sync-data> <ns1-target-user-dn>
cn=GLCANYON,ou=finance,dc=prod,dc=testco,dc=com </ns1-target-user-dn> </
ns1-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. The file is located in the [cred_prov folder](#) on the Identity Manager media.

4 Managing Credential Provisioning






There are additional tasks you can perform to manage the OpenText Credential Provisioning policies after they are implemented.

- ♦ [“Managing the Repository and Application Objects” on page 27](#)
- ♦ [“Managing the Credential Provisioning Policies” on page 27](#)

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.

Option	Description
 Edit	Launches the editor for the resource object.
 Copy	Copies the selected resource object.
Export to Configuration File	Saves the resource object as a .xml file.
 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 selected resource object.
Properties	Lets you rename the selected resource object.

Managing the Credential Provisioning Policies

- 1 In the Outline view, right-click the policy.
- 2 Select the desired task.

Option	Description
 Edit	Launches the Policy Builder.
 Copy	Copies the selected policy.
 Save As	Saves a copy of the policy with another name.

Option	Description
 Simulate	Lets you test the policy in Designer before it is deployed into the Identity Vault.
Export to Configuration File	Saves the resource object as a .xml file.
 Live > Deploy	Deploys the policy into the Identity Vault.
 Live > Compare	Compares the policy to the corresponding object in the Identity Vault.
 Delete	Deletes the selected policy.
Properties	Lets you rename the selected policy.