Novell Identity Manager Driver for Delimited Text

3.5.1

www.novell.com

IMPLEMENTATION GUIDE

March 5, 2008

Novell®

Legal Notices

Novell, Inc.
404 Wyman Street, Suite 500
Waltham, MA 02451
U.S.A.
www.novell.com

Online Documentation: To access the online documentation for this and other Novell products, and to get updates, see Novell Documentation (http://www.novell.com/documentation/).

Contents

About This Guide

This guide explains how to install and configure the Identity Manager Driver for Delimited Text (Delimited Text driver).

Audience

This guide is for Novell® eDirectory and Identity Manager administrators who are using the Delimited Text driver.

Feedback

We want to hear your comments and suggestions about this manual and the other documentation included with this product. Use the User Comment 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 this document, see *Identity Manager Driver for Delimited Text* in the Identity Manager Drivers section on the Novell Documentation Web site (http://www.novell.com/documentation/idm35drivers/index.html).

Additional Documentation

For information on Identity Manager and other Identity Manager drivers, see the Identity Manager Documentation Web site (http://www.novell.com/documentation/idm35/index.html).

Documentation Conventions

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

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

Introducing the Delimited Text Driver

1

1.1 What's New?

The Identity Manager Driver for Delimited Text 3.5.1 has no new features for Identity Manager 3.5.1.

For information on new features in Identity Manager, see "What's New in Identity Manager 3.5.1? " in the *Identity Manager 3.5.1 Installation Guide*.

1.2 Driver Concepts

1.2.1 Java Interfaces to the Driver

The Delimited Text driver includes four Java* interfaces that enable you to add extensions, which are optional. These enhancements to the driver require Java programming. For more information, see "Using Java Interfaces to Customize File Processing" on page 36.

1.2.2 How the Delimited Text Driver Works

The Delimited Text driver uses the Publisher channel, the Subscriber channel, and policies to control data flow.

Publisher and Subscriber Channels

The Delimited Text driver supports the Publisher and Subscriber channels:

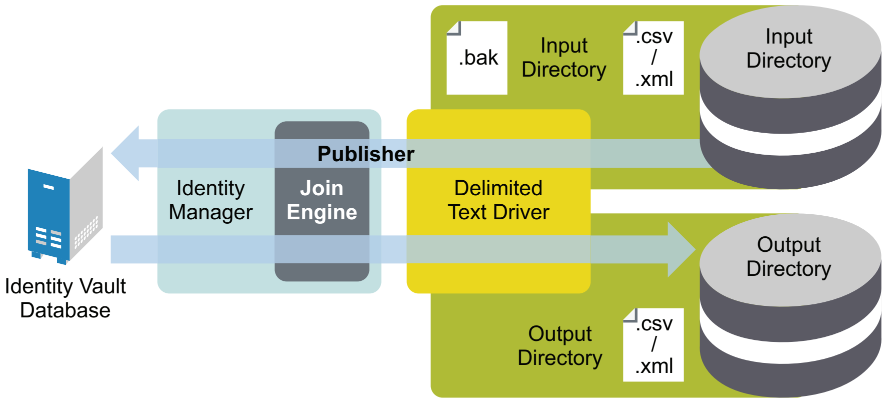
- The Publisher channel reads information from input text files on your local file system and submits that information to the Identity Vault via the Metadirectory engine.

  By default, the Publisher channel does the following:

  1. Checks the input directory every 10 seconds.
  2. Processes any files that have a `.csv` extension.
  3. Changes `.csv` extensions of processed files to `.bak`.
  4. Cycles through this process until you stop the driver.
- The Subscriber channel watches for additions and modifications to Identity Vault objects and creates output files on your local file system that reflect those changes.

By default, the Subscriber channel keeps an output file open until either 200 transactions have been logged or 30 seconds have elapsed. When either of these thresholds is reached, the output file is saved with a *`number`*`.csv` filename and a new output file is opened.

***Figure 1-1** Data Flow*



The example configuration that ships with this driver includes both Subscriber and Publisher channels. However, in many configurations, only one-way data flow is required. In those configurations, only a Publisher or Subscriber channel is used. The other channel is disabled.

Policies

Policies control data synchronization between the driver and the Identity Vault. The following table provides information on the set of preconfigured policies that come with the Delimited Text driver. To customize these policies, use Novell® iManager, as explained in "Configuring the Delimited Text Driver" on page 27.

***Table 1-1** Preconfigured Policies*

Policy	Description
Schema Map	Configured on the driver object. Maps Identity Vault User properties to application attributes as follows: Surname > LastName Given Name > FirstName Title > Title Internet EMail Address > Email Telephone Number > WorkPhone Facsimile Telephone Number > Fax mobile > WirelessPhone Description > Description The application attributes correspond to the sequence of values in the file or, if present, to the attributes associated with unnamed XDS <`field`> elements.

Policy	Description
Input Transform	Configured on the driver object. If the input document is an XML document, no transformations are made. If the document is a delimited text file, each record is transformed into an XDS Add element for User objects with attributes defined by the schema map. The user CN is formed by concatenating the values of first name and last name. Associations are defined by value the user's e-mail attribute.
Output Transform	Configured on the driver object. Specifies that a comma is used as the delimiter character for output files and that the file format is comma-separated value (CSV) format.
Create	Configured on the Publisher channel. Specifies that in order for a User to be created in an Identity Vault, the Given Name and Internet EMail Address attributes must be defined.
Matching	Configured on the Publisher channel. Specifies that a user in an Identity Vault is the same user specified in the input file when the value of Internet Email Address is the same in both places. In the case of a match, only changed attributes are updated in the Identity Vault.
Placement	Configured on the Publisher channel. Specifies that a new user is placed in the Users or Active container and named with the CN created by the Input Transform rule. You need to create a Users\Active container at the root of your tree before you start the driver.
Event Transform	Configured on the Subscriber channel. If an Identity Vault reports a Modify or Sync event, those events are changed to an instance element that can be used to create a complete output record.

1.3 Driver Features

1.3.1 Local and Remote Platforms

The Delimited Text driver runs in any Identity Manager 3.5.1 installation or Remote Loader installation. See "Identity Manager Components and System Requirements" in the *Identity Manager 3.5.1 Installation Guide*.

1.3.2 Role-Based Entitlements

The Delimited Text driver does not implement entitlements.

1.3.3 Password Synchronization

The Delimited Text driver has policies to handle password synchronization. However, no automatic password synchronization exists for the Delimited Text driver. You must be aware of and decide on which attribute you want to hold the password, then synchronize that attribute. Any password synchronization for the driver is just an attribute synchronization.

1.3.4 Synchronizing Data

You can use the Publisher or Subscriber channel to synchronize data.You do this by configuring the driver filter. See Section 5.1.4, "Setting Up One-Way Synchronization," on page 33.

1.3.5 Supported File Types

The example configuration currently supports two types of files:

- "Comma-Separated Values Files" on page 14
- "XML Files in XDS Format" on page 14

Comma-Separated Values Files

Comma-separated value (CSV) files are text files that contain data divided into fields and records. Fields are delimited by commas, and records are delimited by a hard return.

If you need a comma or hard return within the value of a particular field, the entire field value should be enclosed in quotes.

Because the meaning of each field in a CSV file is derived from its position, each record in a CSV file should have the same number of fields. Field values can be left blank, but each record should have the same number of delimiter characters.

XML Files in XDS Format

The XDS format is the defined Novell subset of possible XML formats. This is the initial format for data coming from an Identity Vault. By modifying default rules and changing the style sheets, the Delimited Text driver could be configured to work with any XML format.

For detailed information on the XDS format, refer to NDS DTD Commands and Events (http://developer.novell.com/ndk/doc/dirxml/index.html?dirxmlbk/data/a5323rs.html).

For information on configuring the driver to use XML files in the XDS format, refer to "Configuring for XDS XML Files" on page 34.

Installing the Delimited Text Driver

2

2.1 Upgrading to Identity Manager 3.5.1

You can upgrade from DirXML® 1.1a, Identity Manager 2.*x*, or Identity Manager 3.*x* to Identity Manager 3.5.1.

During an Identity Manager installation, you can install the Delimited Text driver (along with other Identity Manager drivers) at the same time that the Metadirectory engine is installed. See “Installing Identity Manager” in the *Identity Manager 3.5.1 Installation Guide*.

2.2 Where to Install the Driver

You can install the Delimited Text driver locally or remotely.

To install locally, install the driver on the same computer where an Identity Vault and the Metadirectory engine are installed.

If platform or policy constraints make a local configuration difficult or impossible, install the driver on the computer hosting the target application. This installation is referred to as a remote configuration.

Unless you extend this driver’s functionality along with the Java interfaces, it is capable of only reading input files from the local file system of the computer where the driver is running.

2.3 Prerequisites

- ❑ Novell® Identity Manager 3.5.1
- ❑ Designer for Identity Manager, a version of Novell iManager with Identity Manager plug-ins, or ConsoleOne® with plug-ins

2.4 Installing the Driver Separately

This section assumes that you have already installed the Metadirectory engine (and, most likely, other drivers) on the server and need to install only the Delimited Text driver. For instructions on installing Identity Manager, see “Installing Identity Manager” in the *Identity Manager 3.5.1 Installation Guide*.

Typically, an Identity Manager installation installs all drivers, including the Delimited Text driver, at the same time that the Metadirectory engine is installed. If the Delimited Text driver wasn’t installed at that time, you can install the driver separately. The schema won’t be extend during this driver install because the Identity Manager installation already extended it when the Metadirectory engine was installed.

The installation process installs the driver on the server where Identity Manager is installed or on a server where you will use the Remote Loader to run the driver. It also installs the driver information and policies on the iManager server.

Table 2-1 *Where to Find Install Instructions*

Platform	**Where to Find Install Instructions**
Linux	See "Installing Identity Manager through the GUI Interface on UNIX/Linux Platforms" in the *Identity Manager 3.5.1 Installation Guide*
NetWare	See "Installing Identity Manager on NetWare" in the *Identity Manager 3.5.1 Installation Guide*
Windows	See "Installing Identity Manager on Windows" in the *Identity Manager 3.5.1 Installation Guide*.

Upgrading the Delimited Text Driver

3

If you are upgrading from Identity Manager 3.5.0 to Identity Manager 3.5.1, skip this section. The information does not apply.

- Section 3.1, "Upgrading While Installing Identity Manager 3.5.1," on page 17
- Section 3.2, "Upgrading after Identity Manager Is Installed," on page 17

Identity Manager 3.5 contained a new architecture for how policies reference one another. To take advantage of this architecture, the driver configuration file provided for the Delimited Text driver earlier than the Delimited Text 3.5 driver must be upgraded. For more information on this architecture, see "Upgrading Identity Manager Policies" in *Understanding Policies for Identity Manager 3.5.1*.

3.1 Upgrading While Installing Identity Manager 3.5.1

During an Identity Manager 3.5.1 installation, you can install the Delimited Text driver (along with other Identity Manager drivers) at the same time that Identity Manager 3.5.1 is installed. You can upgrade from DirXML® 1.1a, Identity Manager 2, or Identity Manager 3 to Identity Manager 3.5.1.

3.2 Upgrading after Identity Manager Is Installed

If Identity Manager 3.5 1 is already installed, you can use either Designer for Identity Manager or iManager to upgrade the Delimited Text driver.

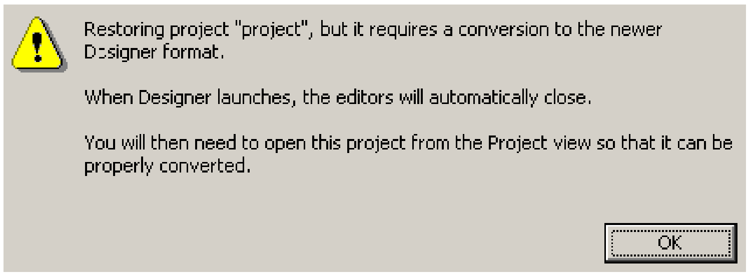
3.2.1 Using Designer to Upgrade

1. Make sure that you have updated your driver with all the patches for the version you are currently running.

   We recommend this step for all drivers, to help minimize upgrade issues.
2. Back up the driver.

   See Chapter 10, "Backing Up the eDirectory Driver," on page 73.
3. Install Designer version 2.1 or later, then launch Designer.

   If you had a project open in Designer when you upgraded Designer, proceed to Step 4. If you didn't have a project open in Designer when you upgraded Designer, skip to Step 5.
4. If you had a project open when upgrading Designer, read the warning message, then click *OK*.



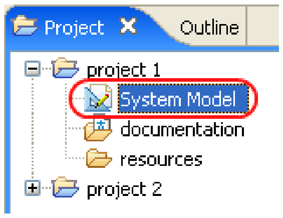
Restoring project "project", but it requires a conversion to the newer Designer format.

When Designer launches, the editors will automatically close.

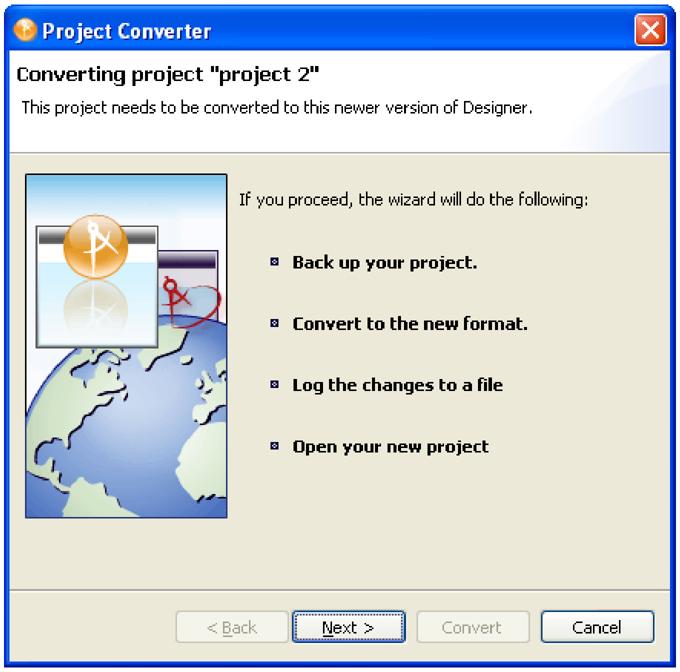
You will then need to open this project from the Project view so that it can be properly converted.

Designer closes the project to perform the upgrade.

5 To open and convert the project, double-click *System Model* in the Project view.



6 Read the tasks listed in the Project Converter message, then click *Next*.



Project Converter

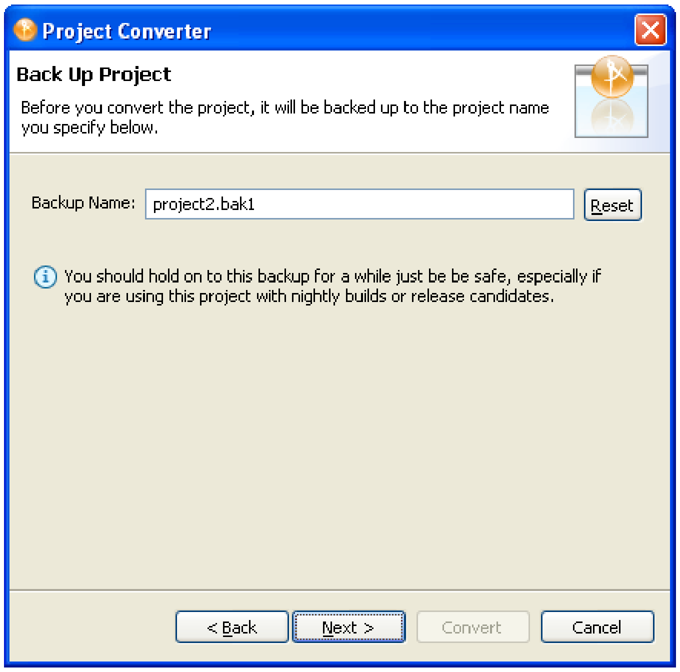
Converting project "project 2"

This project needs to be converted to this newer version of Designer.

If you proceed, the wizard will do the following:

- **Back up your project.**
- **Convert to the new format.**
- **Log the changes to a file**
- **Open your new project**

7 Specify the name of the backup project name, then click *Next*.



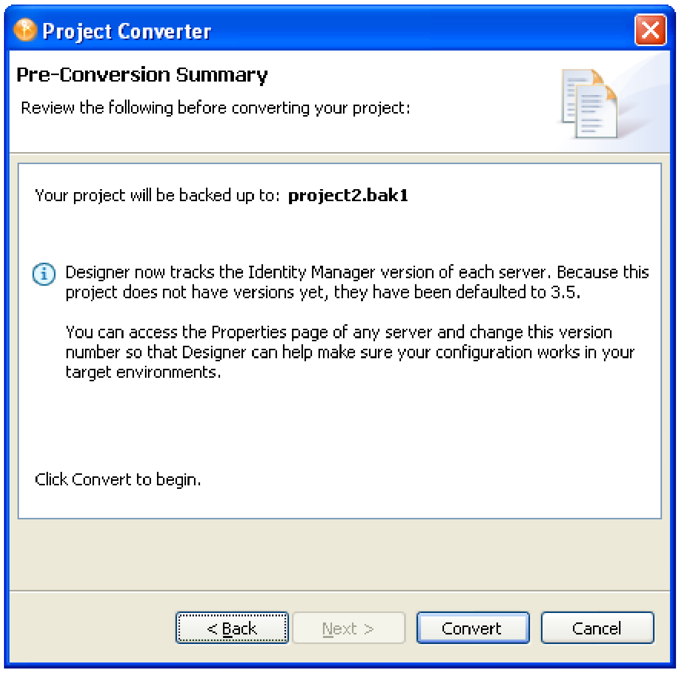
Back Up Project

Before you convert the project, it will be backed up to the project name you specify below.

Backup Name: project2.bak1

You should hold on to this backup for a while just be be safe, especially if you are using this project with nightly builds or release candidates.

8 Read the project conversion summary, then click *Convert*.



Pre-Conversion Summary

Review the following before converting your project:

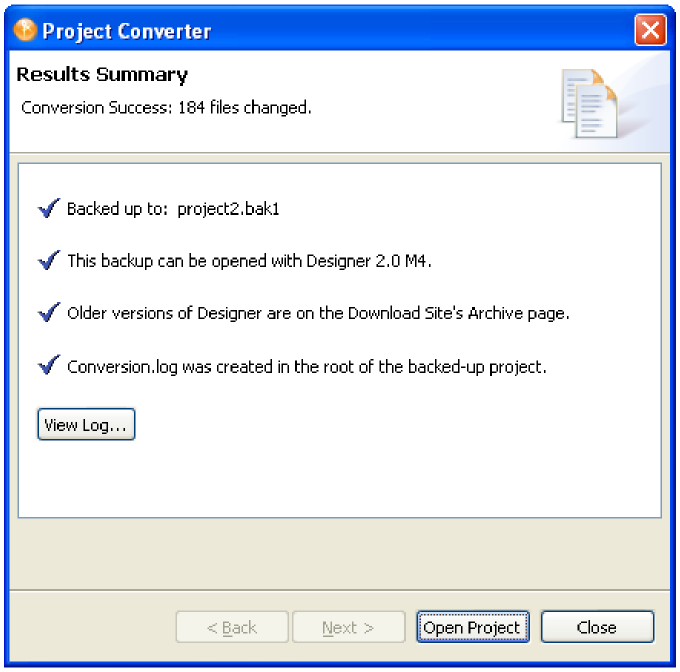
Your project will be backed up to: **project2.bak1**

Designer now tracks the Identity Manager version of each server. Because this project does not have versions yet, they have been defaulted to 3.5.

You can access the Properties page of any server and change this version number so that Designer can help make sure your configuration works in your target environments.

Click Convert to begin.

9 Read the project conversion result summary, then click *Open Project*.


Project Converter

Results Summary

Conversion Success: 184 files changed.

✓ Backed up to: project2.bak1

✓ This backup can be opened with Designer 2.0 M4.

✓ Older versions of Designer are on the Download Site's Archive page.

✓ Conversion.log was created in the root of the backed-up project.

To view the log file that is generated, click *View Log*.

3.2.2 Using iManager to Upgrade

1 Make sure that you have updated your driver with all the patches for the version you are currently running.

   To help minimize upgrade issues, we recommend that you complete this step on all drivers.

2 Back up the driver.

   See Chapter 10, “Backing Up the eDirectory Driver,” on page 73.

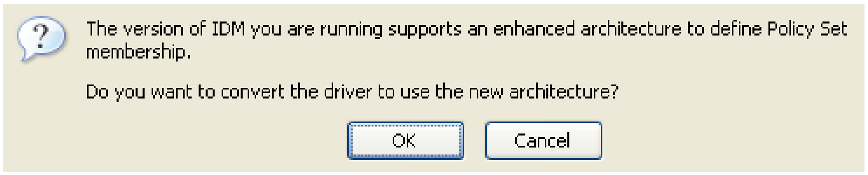
3 Verify that Identity Manager 3.5.1 has been installed and that you have the current plug-ins installed.

4 Launch iManager.

5 Click *Identity Manager* > *Identity Manager Overview*.

6 Click *Search* to find the Driver Set object, then click the driver that you want to upgrade.

7 Read the message that is displayed, then click *OK*.


The version of IDM you are running supports an enhanced architecture to define Policy Set membership.

Do you want to convert the driver to use the new architecture?

IMPORTANT: The example configuration file for the updated driver changed for the Identity Manager 3.0 release. If your current configuration meets your requirements, you don’t need to import this example configuration. If you *do* import the new example configuration, you will see an additional driver for Delimited Text with a new name, a new Identity Vault container specified in the placement rule, and new rule names.

Importing an Example Delimited Text Configuration File 4

The Delimited Text driver includes an example configuration file that you can use as a starting point for creating the Driver object. When you import this file, Designer for Identity Manager or iManager creates and configures the objects and policies needed to make the driver work properly.

- Section 4.1, "Preparing Data Locations," on page 23
- Section 4.2, "Using Designer to Import," on page 23
- Section 4.3, "Using iManager to Import," on page 24
- Section 4.4, "Configuration Parameters: Delimited Text," on page 25

4.1 Preparing Data Locations

If you use all of the defaults provided by the example configuration, prepare locations for Identity Manager data.

1 Add the containers Users\Active at the root level of your Identity Vault, then do one of the following:

2 Create directories.

   - On Windows*, point to or create an input directory and an output directory on your local file system.

     For example, point to the two directories created in Step 1 on page 24 and Step 2 on page 24. You can use any two directories on your local file system.

   - On Solaris, Linux*, or NetWare®, create input and output directories wherever you'd like. Then update the driver configuration with the correct platform-specific paths. For more information, refer to "Input File Path" on page 32.

4.2 Using Designer to Import

You can use Designer to import the basic driver configuration file for the Delimited Text driver by using Designer. This basic file creates and configures the objects and policies needed to make the driver work properly.

The following procedure explains one of several ways to import the example configuration file:

1 Open a project in Designer.

2 In the Modeler, right-click the Driver Set object, then select *New* > *Driver*.

3 From the drop-down list, select *Delimited Text-CSV*, then click *Run*.

4 Configure the driver by filling in the fields.

   Specify information specific to your environment. See the ettings and values in Table 4-1 on page 25.

5 After specifying parameters, click *Finish* to import the driver.

6 Test the driver.

7 Deploy the driver into the Identity Vault.

See "Deploying a Driver to an Identity Vault" in the *Designer 2.1 for Identity Manager 3.5.1* guide.

4.3 Using iManager to Import

Identity Manager provides an example configuration file. You installed this file when you installed the Identity Manager Web components on an iManager server. Think of the example configuration file as a template that you import and customize or configure for your environment.

1 Create a local directory for output files.

For example, on Windows create `c:\csvsample\output`.

This directory can be any directory on your local file system. Files on this driver's Subscriber channel are created and placed here.

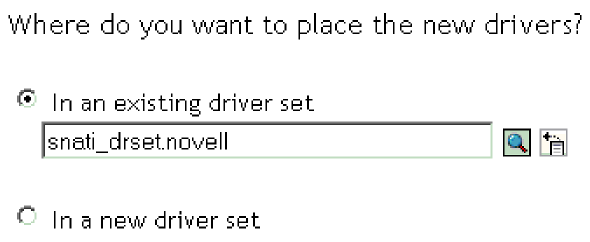
2 Create a local directory where input files can be created.

For example, on Windows create `c:\csvsample\input`.

This directory can be any directory on your local file system. Files for this driver's Publisher channel are placed here. The driver looks in this directory for files to process.

3 In iManager, select *Identity Manager Utilities* > *Import Configurations*.

4 Search for and select a driver set, then click *Next*.



Where do you want to place the new drivers?

● In an existing driver set

snati_drset.novell

○ In a new driver set

If you place this driver in a new driver set, you must specify a driver set name, context, and associated server.

5 Select *Delimited Text - CSV*, then click *Next*.

6 Configure the driver by filling in the configuration parameters.

For information on the settings, see the settings and values in Table 4-1 on page 25.

7 Define security equivalences by using a User object that has the rights that the driver needs to have on the server

The Admin user object is most often used for this task. However, you might want to create a DriversUser (for example) and assign security equivalence to that user. Whatever rights that the driver needs to have on the server, the DriversUser object must have the same security rights.

8 Identify all objects that represent administrative roles and exclude them from replication.

Exclude the security-equivalence object (for example, DriversUser) that you specified in Step 2. If you delete the security-equivalence object, you have removed the rights from the driver. Therefore, the driver can't make changes to Identity Manager.

9 Click *Finish*.

4.4 Configuration Parameters: Delimited Text

The following table lists basic parameters that you set while using the Driver Configuration Wizard. For additional parameters, see Appendix B, "Properties of the Delimited Text Driver," on page 91.

Table 4-1 *Delimited Text Settings*

Field	Description
Driver Name	The object name to be assigned to this driver.
Output File Path	Specify the platform-specific path to the local directory where output files should be created. This path is to the directory that you created in Step 1 on page 24.
Output File Extension	Specify the extension to append to output files when the files are created.
Input File Path	Specify the platform-specific path to the local directory where input files exist. This path is to the directory that you created in Step 2 on page 24.
Input File Extension	Specify the extension used to designate input files.
Rename File Extension	Specify the extension that an input file will be renamed with after the file has been processed. Leave this field blank if you want the file deleted.
New User Container	Specify the DN of the container where new users are to be placed.
Configure Data Flow	◆ **Bi-Directional:** Delimited Text and an Identity Vault are both authoritative sources of the data synchronized between them. ◆ **Delimited Text to Identity Vault:** Delimited Text is the authoritative source. ◆ **Identity Vault to Delimited Text:** The Identity Vault is the authoritative source.
Password Failure Notification User	Optionally, specify an additional user (for example, admin) to receive notification when a user's password fails.
Driver Is Local/Remote	Configure the driver for use with the Remote Loader service by selecting the Remote option, or select Local to configure the driver for local use. If you select Local, you can skip the remaining parameters.
Remote Host Name and Port	Specify the host name or IP address and port number for where the Remote Loader service has been installed and is running for this driver. The default port is 8090.
Driver Password	The driver object password is used by the Remote Loader to authenticate itself to the Metadirectory server. It must be the same password that is specified as the driver object password on the Identity Manager Remote Loader.
Remote Password	The Remote Loader password is used to control access to the Remote Loader instance. It must be the same password that is specified as the Remote Loader password on the Identity Manager Remote Loader.

Configuring the Delimited Text Driver

5

The Delimited Text driver includes an example configuration that you can use as a starting point for your deployment. Most deployments require you to change the example configuration.

For example, if you need only one-way data synchronization, or if the attributes you're synchronizing are different from the eight provided in the example file, you need to customize the driver.

- "Configuring Driver Parameters" on page 27
- "Configuring Data Synchronization" on page 35
- "Using Java Interfaces to Customize File Processing" on page 36

NOTE: When you customize data synchronization, you must work within the supported standards and conventions for the operating systems and accounts being synchronized. Data containing characters that are valid in one environment, but invalid in another, causes errors.

5.1 Configuring Driver Parameters

When you change driver parameters, you tune driver behavior to align with your network environment. For example, you might find the default Publisher polling interval to be shorter than your synchronization requires. Making the interval longer could improve network performance while still maintaining appropriate synchronization.

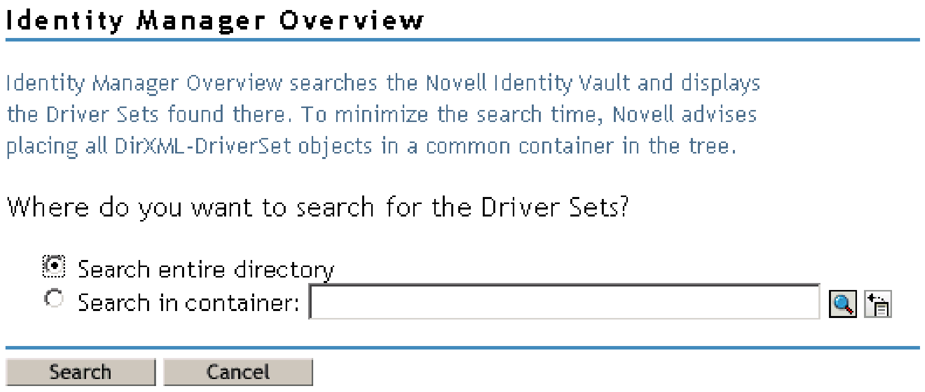
Driver parameters are divided into the following settings:

- Driver (See "Driver Settings" on page 29.)
- Subscriber (See "Subscriber Settings" on page 30.)
- Publisher (See "Publisher Settings" on page 32.)

To configure the driver parameters:

1. In iManager, select *Identity Manager > Identity Manager Overview*.
2. Search for or browse to and select a driver set.

   The following figure illustrates using the Search feature.



Identity Manager Overview

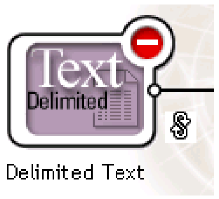
Identity Manager Overview searches the Novell Identity Vault and displays the Driver Sets found there. To minimize the search time, Novell advises placing all DirXML-DriverSet objects in a common container in the tree.

Where do you want to search for the Driver Sets?

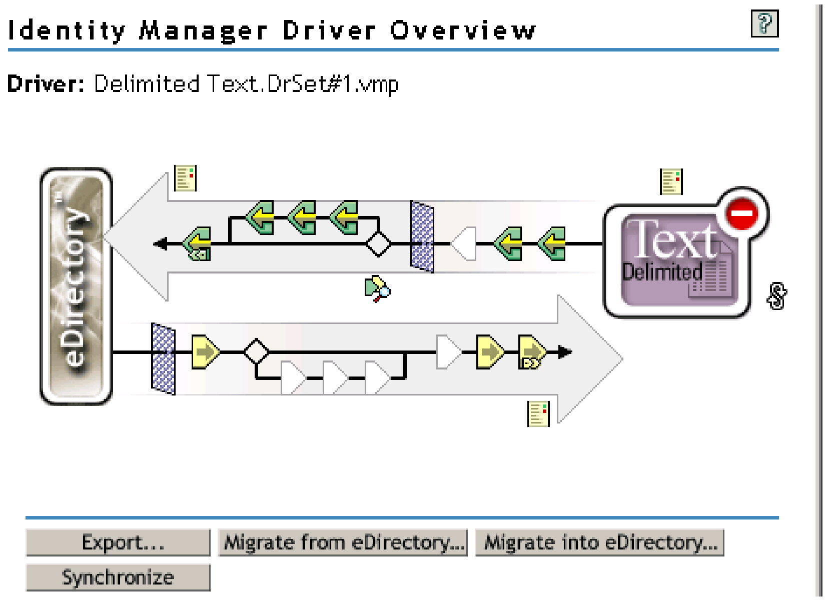
[✓ Search entire directory]

Search in container:

3 View the driver overview by clicking the driver icon (Delimited Text).



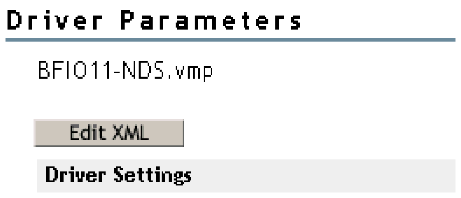
4 Access the Driver Configuration page by clicking the driver icon again.



Identity Manager Driver Overview

Driver: Delimited Text.DrSet#1.vmp

5 Scroll to the *Driver Parameters* section, make changes, then click *OK*.



Driver Parameters

BFI011-NDS.vmp

Driver Settings

To make changes to the *Driver Settings* section, see "Driver Settings" on page 29.

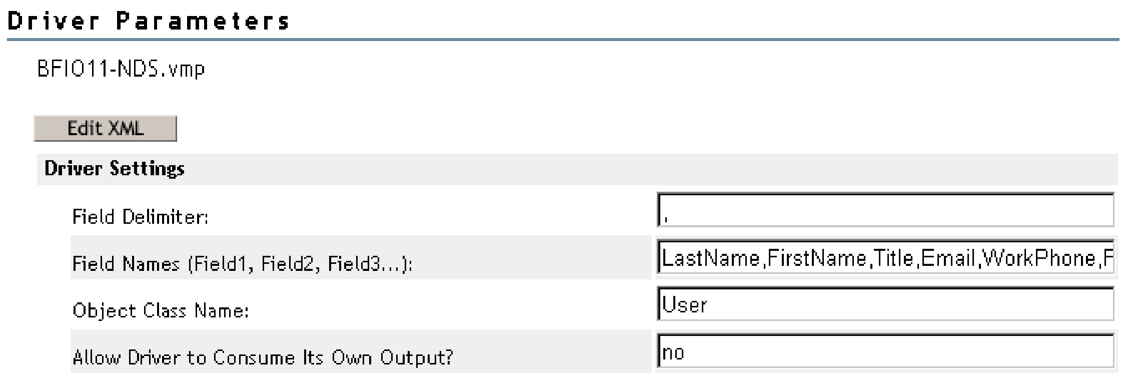
To make changes to the *Subscriber Settings* section, see "Subscriber Settings" on page 30.

To make changes to the *Publisher Settings* section, see "Publisher Settings" on page 32.

5.1.1 Driver Settings

The following figure illustrates the driver parameters and their default values in the example configuration.

Figure 5-1 *Driver Parameters*



Driver Parameters

BFIO11-NDS.vmp

Driver Settings

Field Delimiter:	,
Field Names (Field1, Field2, Field3...):	LastName,FirstName,Title,Email,WorkPhone,F
Object Class Name:	User
Allow Driver to Consume Its Own Output?	no

Field Delimiter

Field Delimiter indicates the character that is used to delimit field values in the input files. It must be one character.

If the values of any of the input fields contain this character, enclose the entire value in quotes to prevent it from being seen as a delimiter.

NOTE: Changing this delimiter parameter to something other than a comma does not automatically change the delimiter character used in the output files when a Subscriber is used. To change the delimiter character in the output files, edit the Output Transform style sheet. The delimiter character is assigned to a variable near the top of that style sheet.

Field Names

Field Names is a comma-separated list of attribute names that can be referred to in the Schema Mapping rule. In the input files, the fields of the records must correspond to the order and positioning of the names in this list.

For example, if you list eight field names in this parameter, each record of the input files should have eight fields separated by the field delimiter character. On NetWare® and Windows, see `sample.csv` in the `delimitedtext/samples` directory for an example. On Solaris and Linux, `sample.csv` is located in the `/usr/lib/dirxml/rules/delim` directory.

The following table lists the default values:

Table 5-1 Default Values for Field Names

Parameter	Sample Configuration Value
Field Names (Field 1, Field 2, Field 3...)	LastName,FirstName,Title,Email,WorkPhone,Fax, WirelessPhone,Description

Object Class Name

Object Class Name is the Novell® eDirectory™ class name that should be used when creating new objects to correspond to input files.

Allow Driver to Consume Its Own Output

This parameter prevents you from inadvertently creating a situation in which the driver writes output files that are immediately read in again as input of the same driver.

The default is No. By default, the driver won't load if all the following conditions occur:

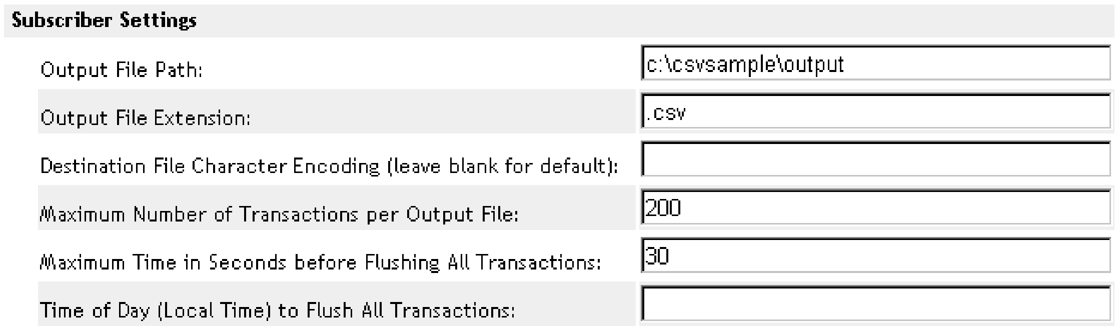
- You have both a Subscriber channel and a Publisher channel.
- The input and output directories are the same.
- The input and output file extensions are the same.

If you want to feed the output of the Subscriber channel into the input of the Subscriber channel as a way to detect Identity Vault events to trigger other changes in the Identity Vault, set this parameter to Yes. For example, to update the Full Name attribute when the Given Name, Surname, or Initials attributes are updated, set this parameter to Yes.

5.1.2 Subscriber Settings

The following figure illustrates the Subscriber settings and their default values in the example configuration.

Figure 5-2 Subscriber Channel Settings



Subscriber Settings

Output File Path:	c:\csvsample\output
Output File Extension:	.csv
Destination File Character Encoding (leave blank for default):	
Maximum Number of Transactions per Output File:	200
Maximum Time in Seconds before Flushing All Transactions:	30
Time of Day (Local Time) to Flush All Transactions:	

Output File Path

Output File Path is the directory on the local file system where output files will be created. An error occurs if this directory doesn't exist.

Table 5-2 *Example Configuration Values*

Platform	**Example Configuration Value**
Windows	`c:\csvsample\output`
Solaris or Linux	`/csvsample/output`
NetWare	Specify the volume (for example, `sys:csvsample\output`)

Output File Extension

Output files have a unique name that ends with the characters in the *Output File Extension* parameter. If the output files from a Subscriber channel are used as input files for the Publisher channel of another Delimited Text driver, the destination file extension must match the source file extension parameter of the second driver.

Destination File Character Encoding

When the *Destination File Character Encoding* parameter contains no value, the default Java character encoding for your locale is used.

To use an encoding other than the default for your locale, enter one of the canonical names from the Supported Encodings table (http://java.sun.com/j2se/1.4.2/docs/guide/intl/encoding.doc.html).

NOTE: The Publisher and Subscriber channels can use different character encodings.

Maximum Number of Transactions Per Output File

This parameter determines the maximum number of transactions that are written to a single output file. When the file transaction limit is reached, the file closes, and a new file is created for subsequent transactions. To limit the number of transactions that can be written to a single file, leave this parameter blank or set it to zero.

For more information, see "Maximum Time in Seconds Before Flushing All Transactions" on page 31.

Maximum Time in Seconds Before Flushing All Transactions

If no new transactions have been written to the output file in the amount of time specified in this parameter, the file is closed. When new transactions need to be written, a new output file is created. If you don't want to limit the time that can pass before the output file is closed, leave this parameter blank or set it to zero.

Time of Day (Local Time) to Flush All Transactions

If a value is supplied for this parameter, the current output file is closed at the specified time each day. Subsequent transactions are written to a new file. This parameter does not prevent the *Maximum Number of Transactions per Output File* or the *Maximum Time in Seconds before Flushing All Transactions* parameters from also acting as output file thresholds. If you use this parameter and only want one file per day, set the other two parameters to zero.

The format of this parameter can be HH:MM:SS (using the 24-hour clock) or H:MM:SS AM/PM. An hour is required, but the minutes and seconds are optional. Because the parameter assumes local time, any time zone information included in the value is ignored.

NOTE: The previous three parameters (*Maximum Number of Transactions per Output File*, *Maximum Time in Seconds before Flushing All Transactions*, and *Time of Day to Flush All Transactions*) are all capable of acting as a threshold for the transaction size a file is able to grow to, or for the time that it remains open to accept new transactions.

As long as an output file is still open for writing by the Delimited Text driver, it shouldn't be considered as finalized. Avoid opening the file in any other process until the driver closes the file. For this reason, one of the three previous parameters must be set to assure that output files don't remain open indefinitely. To avoid this condition, if the driver detects that all three parameters are blank (or zero), it automatically sets the *Maximum Number of Transactions per Output File* to the value of 1.

5.1.3 Publisher Settings

The following table lists the Publisher channel parameters and their default values in the example configuration.

Table 5-3 *Publisher Channel Settings*

Setting	Sample Configuration Value
Input File Path	On Windows: `c:\csvsample\input` On Solaris and Linux: `/usr/lib/dirxml/rules/delim` For NetWare, you need to specify the volume (for example, `sys:csvsample\input`)
Input File Extension	`.csv`
Source File Character Encoding (leave blank for default)	[blank]
Rename File Extension (leave blank to delete a file)	`.bak`
Polling Rate (in seconds)	10

Input File Path

The Publisher channel looks for new input files in the *Input File Path*, which is a directory on the local file system.

Input File Extension

The Publisher channel uses only files that have the extension specified in this parameter. After the files have been processed, the value of the *Rename File Extension* parameter is appended to the filename, so the Publisher channel won't try to process the same file again. If the value of the *Rename File Extension* parameter is left blank, the source file is deleted after it is processed.

Source File Character Encoding

When the *Source File Character Encoding* parameter contains no value, the default Java character encoding for your locale is used.

To use an encoding other than the default for your locale, enter one of the canonical names from the Supported Encodings table (http://java.sun.com/j2se/1.4.2/docs/guide/intl/encoding.doc.html).

If the Input File Extension parameter is `.xml,` the *Source File Character Encoding* can be indicated in one of two ways:

- If a value is indicated in the *Source File Character Encoding* parameter, it is used.
- If the parameter is blank, and if the XML document specifies an Encoding Declaration as described in the W3C XML Recommendation (http://www.w3.org/TR/REC-xml#charencoding) in paragraph 4.3.3, the Encoding Declaration is handled by the XML parser in the Metadirectory engine.

  The Identity Manager XML parser handles the following character encodings:

  - UTF-8
  - UTF-16
  - ISO-8859-1
  - US-ASCII

NOTE: The Publisher and Subscriber channels can use different character encodings.

Rename File Extension

The Publisher channel uses only files that have the extension specified in the parameter. After the files have been processed, the value of the *Rename File Extension* parameter is appended to the filename, so the Publisher channel won't try to process the same file again. If the value of the *Rename File Extension* parameter is left blank, the source file is deleted after it is processed.

IMPORTANT: If you change the default, use only characters that are valid in filenames on your platform. Invalid characters cause the rename to fail and the driver to reprocess the same file repeatedly.

Polling Rate

When the Publisher channel has finished processing all source files, it waits the number of seconds specified in this parameter before checking for new source files to process.

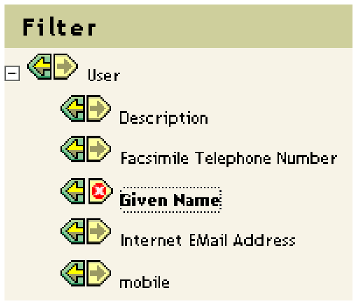
5.1.4 Setting Up One-Way Synchronization

If your data synchronization goes only one way, disable the channel that you won't use. To disable one of the channels, clear the filters on the channel you don't need and don't specify a path for the input or output directory, depending on the channel.

For example, if you only need a Publisher channel:

1 On the Filter editor in iManager, clear the filters on the Subscriber object.

  1a For example, select the *Given Name* filter.

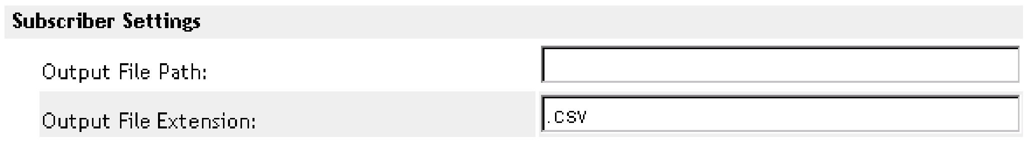


1b Select *Ignore* in the Subscribe section.

As the following figure illustrates, the filter's Subscribe function is disabled.

2 Save the changes by clicking *OK*.

3 In the *Driver Parameters* section, scroll to *Subscriber Settings* and remove the path specified for the *Output File Path*.



Subscriber Settings

Output File Path:	
Output File Extension:	.csv

If you only need a Subscriber channel, clear the filters on the Publisher object and remove the path specified for the *Input File Path* in the *Driver Parameters* section.

5.1.5 Configuring for XDS XML Files

You can use XML files in XDS format instead of comma-separated value (CSV) files with the driver.

Because you generally use this driver only with a Publisher or Subscriber channel, perform only the steps from the section that you need.

Using the Publisher Channel

To have the driver accept input in XML format, change the input file extension to `.xml`.

Using the Subscriber Channel

To have the driver send output in XDS format, remove the Event Transform and Output Transform style sheets from the Subscriber channel.

1 In iManager, select *eDirectory Administration* > *Delete Object*.

2 Browse to the driver's Subscriber object, then select the SubscriberEventTransformSS object.

3 Click *OK*.

4 Click *Repeat Task*.

5 Browse to and select the driver's OutputTransformSS object.

6 Click *OK* twice.

5.2 Configuring Data Synchronization

The real power of Identity Manager is in managing the shared data itself. This section covers some common customizations for the Delimited Text driver.

The example configuration available with the driver uses comma-separated value files. However, you can use the driver in many ways. It is designed to be as flexible as possible. The driver passes the text-based files largely unchanged to the style sheets. The style sheets do most of the work. You can write new style sheets to allow the driver to work with almost any text-based file that contains predictably repeatable patterns.

The basis for this exchange is the `<delimited-text>` XML element. For example, to design a Publisher channel that reads information from a text file, create an Input Transform style sheet that receives the contents of the file and converts it into a `<delimited-text>` element.

The following is an example of a `<delimited-text>` element:

```
<delimited-text>
     <record>
          <field>John</field>
          <field>Maxfield</field>
          <field>555-1212</field>
     </record>
     <record>
          <field>Sarah</field>
          <field>Lopez</field>
          <field>555-3434</field>
     </record>
</delimited-text>
```

When field elements appear like this without an identifying name attribute, the driver uses the field position and matches it with the position of the Field Name driver parameter.

You can provide the field name within the XML:

```
<delimited-text>
     <record>
          <field name="FirstName">John</field>
          <field name="LastName">Maxfield</field>
          <field name="Phone">555-1212</field>
     </record>
     <record>
          <field name="FirstName">Sarah</field>
          <field name="LastName">Lopez</field>
          <field name="Phone">555-3434</field>
     </record>
</delimited-text>
```

For detailed information on writing style sheets to handle other document types, refer to the sample style sheets that come with this driver. If you create the driver by using the example configuration, you can use Input Transform, Output Transform, and Event Transformation style sheets as a starting point.

5.3 Using Java Interfaces to Customize File Processing

- Section 5.3.1, "Four Java Interfaces to the Driver," on page 36
- Section 5.3.2, "Creating a Java Class," on page 37
- Section 5.3.3, "Creating a Java .jar File," on page 37
- Section 5.3.4, "Configuring the Driver to Use the New Class," on page 37

5.3.1 Four Java Interfaces to the Driver

Java interfaces enable you to customize file processing by using Java classes that you write:

- InputSorter
- InputSource
- PreProcessor
- PostProcessor

These enhancements to the driver require Java programming. To implement this functionality, complete the following processes:

- Create a Java class that implements one of the new interfaces
- Create a Java `.jar` file that contains your new class
- Configure the driver to use the new class

These interfaces enable you to add extensions, which are optional. The driver continues to function as before without extensions. However, if you want to directly modify the behavior of the driver, but have been unable to make these modifications from a style sheet or DirXML® Script, extending the Delimited Text driver can be useful.

By using Java classes that you write, you can use the interfaces to customize the publish and subscribe processes in the following ways:

Table 5-4 *Customizing the Publish and Subscribe Processes*

Process	Interface	Description
Publish	InputSorter	Defines the processing order of multiple input files. The system where your driver is installed determines the default processing order. For example, files on an NT system are processed in alphabetical order. You can use the InputSorter to impose the processing order that you require.
Publish	InputSource	Provides data other than the files in the default location for the driver to process. For example, you could check an FTP server for input files and then transfer the files to the local file system for processing.

Process	Interface	Description
Publish	PreProcessor	Ties data manipulation required to prepare input files for driver processing directly to the driver. Before this interface was available, preprocessing was independent of the driver. You could create a separate application that would monitor another directory for input files, modify the files in some way, and then copy the files to the input directory of the driver. By creating a class that implements the PreProcessor, you can do this type of preprocessing more directly.
Subscribe	PostProcessor	Ties data manipulation required by the application consuming Identity Vault output directly to the driver.

5.3.2 Creating a Java Class

JavaDoc and an example class are included with the driver to help you implement this functionality. Find these files at
`platform \dirxml\drivers\delimitedtext\extensions`.

5.3.3 Creating a Java .jar File

After you have implemented your class file, create a Java `.jar` file (Java archive) using the jar tool. The `.jar` file must contain the class that you have created. Put the `.jar` file into the `novell/nds/lib` directory. The path might differ, depending on the platform you're on, but it should be the same location as `DelimitedTextShim.jar` and `DelimitedTextUtil.jar`.

5.3.4 Configuring the Driver to Use the New Class

After you have placed the new `.jar` file in the correct location, configure the driver to use your new class by modifying the driver's properties.

1. In iManager, select *Identity Manager* > *Identity Manager Overview*.
2. Locate the driver in its driver set.
3. Click the driver icon to open the *Identity Manager Driver Overview* page.
4. Click the driver icon again to open the *Modify Object* page.
5. In the drop-down menu, select *Driver Configuration*.
6. Scroll to D*river Parameters*, then click *Edit XML*.
7. Locate the `<publisher-options>` section of the file.

   This file defines which parameters and values appear in the Driver Parameters section of the D*river Configuration* page.

   For each class you created that works on the Publisher channel, you enter an additional option in the `<publisher-options>` section. After you've updated this file, you'll see your new options in the interface.
8. For each new class you created on the Publisher channel, add an entry corresponding to the interface type. Use the following table as a guide:

Interface	New Entry

InputSorter	`<input-sorter display-name="InputSorter Class">com.acme.MyNewClass</input-sorter>`
	
	`<input-sorter-params display-name="InputSorter init string">MY CONFIG PARAMS</input-sorter-params>`

InputSource	`<input-source display-name="InputSource Class">com.acme.MyNewClass</input-source>`
	
	`<input-source-params display-name="InputSource init string">MY CONFIG PARAMS</input-source-params>`

PreProcessor	`<pre-processor display-name="PreProcessor Class">com.acme.MyNewClass</pre-processor>`
	
	`<pre-processor-params display-name="PreProcessor init string">MY CONFIG PARAMS</pre-processor-params>`

8a Replace `com.acme.MyNewClass` with the name of the class that you have defined along with a full package identifier.

8b Replace MY CONFIG PARAMS*MY CONFIG PARAMS* with any information that you want to pass to the init method of your class.

The init method of your class is then responsible for parsing the information contained in this string. If your class doesn't require a configuration string to be passed to the init method, you can leave off the whole element, in which case null would be passed to the init method.

9 If you created a PostProcessor rule, locate the `<subscriber-options>` section of the file and add the following lines:

```
<post-processor display-name="PostProcessor
Class">com.acme.MyNewClass</post-processor> <post-processor-params
display-name="PostProcessor init string">MY CONFIG PARAMS</post-
processor-params>
```

9a Replace `com.acme.MyNewClass` with the name of the class that you have defined along with full package information.

9b Replace `MY CONFIG PARAMS` with any information that you want to pass to the init method of your class.

The init method of your class is then responsible for parsing the information contained in this string. If your class doesn't require a configuration string to be passed to the init method, you can leave off the entire element, in which case null would be passed to the init method.

10 Click *OK*.

6 Activating the Driver

Activate the driver within 90 days of installation. Otherwise, the driver will stop working.

For information on activation, refer to "Activating Novell Identity Manager Products" in the *Identity Manager 3.5.1 Installation Guide*.

Managing the Delimited Text Driver

7

7.1 Starting, Stopping, or Restarting the Delimited Text Driver

In Designer for Identity Manager:

1 Open a project in the Modeler, then right-click the driver line.

2 Select *Live* > *Start Driver*, *Stop Driver,* or *Restart Driver*.

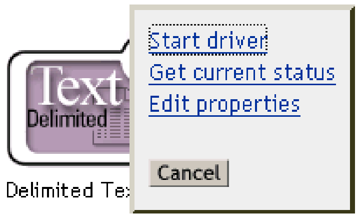
In iManager:

1 If you changed default data locations during configuration, ensure that the new locations exist before you start the driver.

2 Click *Identity Manager* > *Identity Manager Overview*.

3 Browse to or search for the driver set where the driver exists.

4 Click the driver status indicator in the upper right corner of the driver icon, then click *Start driver*, *Stop driver*, or Re*start driver*.



If a change log is available, the driver processes all the changes in the change log. To force an initial synchronization, see "Migrating and Resynchronizing Data" on page 42.

7.2 Migrating and Resynchronizing Data

Identity Manager synchronizes data when the data changes. If you want to synchronize all data immediately, you can choose from the following options:

- **Migrate Data from Identity Vault:** Allows you to select containers or objects you want to migrate from the Identity Vault to an application. When you migrate an object, the Identity Manager engine applies all of the Matching, Placement, and Create policies, as well as the Subscriber filter, to the object.
- **Migrate Data into Identity Vault:** Assumes that the remote application (usually a Web Service) can be queried for entries that match the criteria in the Publisher filter.
- **Synchronize:** The Identity Manager engine looks in the Subscriber class filter and processes all objects for those classes. Associated objects are merged. Unassociated objects are processed as Add events.

To use one of the options explained above:

1. In iManager, click *Identity Manager > Identity Manager Overview*.
2. Browse to and select the driver set where the driver exists, then click *Search*.
3. Click the driver icon.
4. Click the appropriate migration button.

For more information, see Chapter 8, "Synchronizing Objects," on page 57.

7.3 Using the DirXML Command Line Utility

The DirXML® Command Line utility provides command line access to manage the driver. This utility is not a replacement for iManager or Designer. The primary use of this utility is to allow you to create platform-specific scripts to manage the driver.

For example, you could create a shell script on Linux to check the status of the driver. See Appendix A, "The DirXML Command Line Utility," on page 77 for information about the DirXML Command Line utility. For daily tasks, use iManager or Designer.

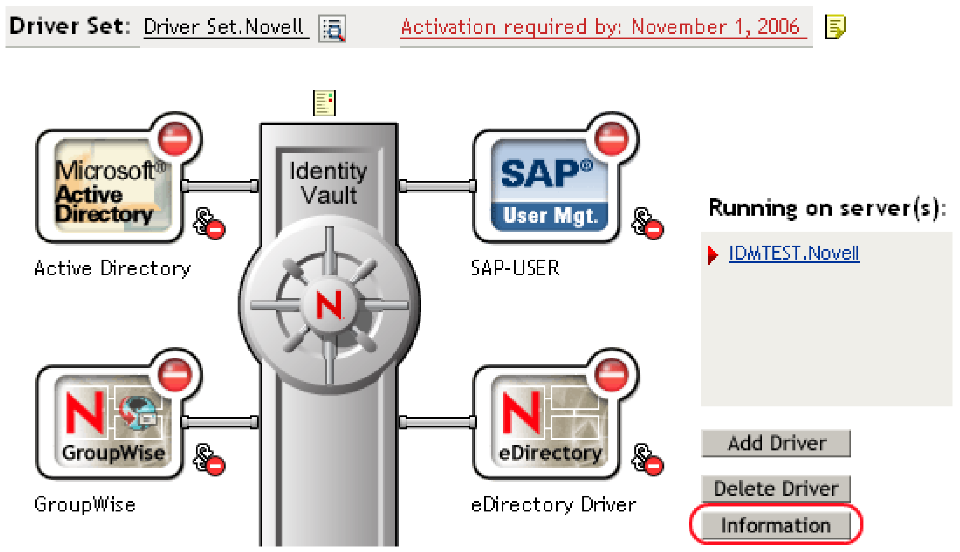
7.4 Viewing Driver Version Information

The Versioning Discovery tool only exists in iManager.

- Section 7.4.1, "Viewing a Hierarchical Display of Version Information," on page 42
- Section 7.4.2, "Viewing the Version Information As a Text File," on page 44
- Section 7.4.3, "Saving Version Information," on page 46

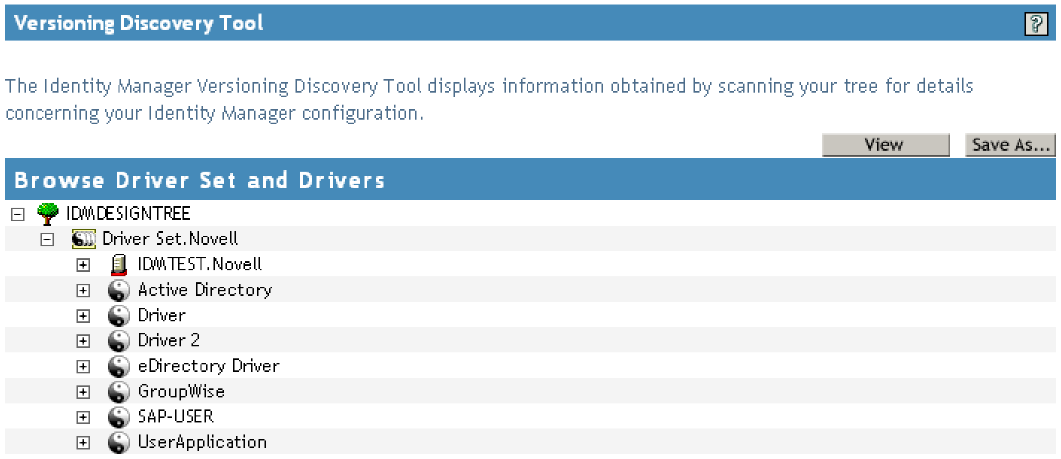
7.4.1 Viewing a Hierarchical Display of Version Information

1. To find your Driver Set object in iManager, click *Identity Manager > Identity Manager Overview*, then click *Search*.
2. In the Identity Manager Overview, click *Information.*



You can also select *Identity Manager Utilities* > *Versions Discovery*, browse to and select the Driver Set object, then click *OK*.

3 View a top-level or unexpanded display of versioning information.


Versioning Discovery Tool

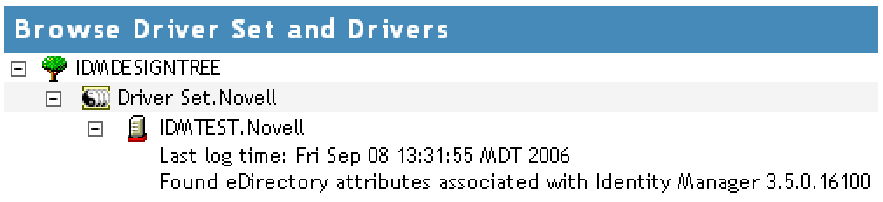
The Identity Manager Versioning Discovery Tool displays information obtained by scanning your tree for details concerning your Identity Manager configuration.

The unexpanded hierarchical view displays the following:

- The eDirectory™ tree that you are authenticated to
- The Driver Set object that you selected
- Servers that are associated with the Driver Set object

  If the Driver Set object is associated with two or more servers, you can view Identity Manager information on each server.
- Drivers

4 View version information related to servers by expanding the server icon.

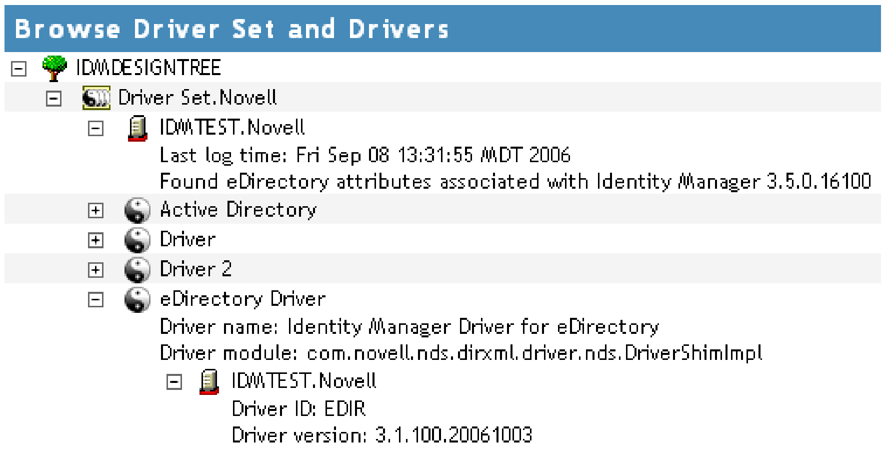


```
IDMDESIGNTREE
  Driver Set.Novell
    IDMTEST.Novell
      Last log time: Fri Sep 08 13:31:55 MDT 2006
      Found eDirectory attributes associated with Identity Manager 3.5.0.16100
```

The expanded view of a top-level server icon displays the following:

- Last log time
- Version of Identity Manager that is running on the server

5 View version information related to drivers by expanding the driver icon.



```
IDMDESIGNTREE
  Driver Set.Novell
    IDMTEST.Novell
      Last log time: Fri Sep 08 13:31:55 MDT 2006
      Found eDirectory attributes associated with Identity Manager 3.5.0.16100
    Active Directory
    Driver
    Driver 2
    eDirectory Driver
      Driver name: Identity Manager Driver for eDirectory
      Driver module: com.novell.nds.dirxml.driver.nds.DriverShimImpl
        IDMTEST.Novell
          Driver ID: EDIR
          Driver version: 3.1.100.20061003
```

The expanded view of a top-level driver icon displays the following:

- The driver name
- The driver module (for example, com.novell.nds.dirxml.driver.delimitedtext.DelimitedTextDriver)

The expanded view of a server under a driver icon displays the following:

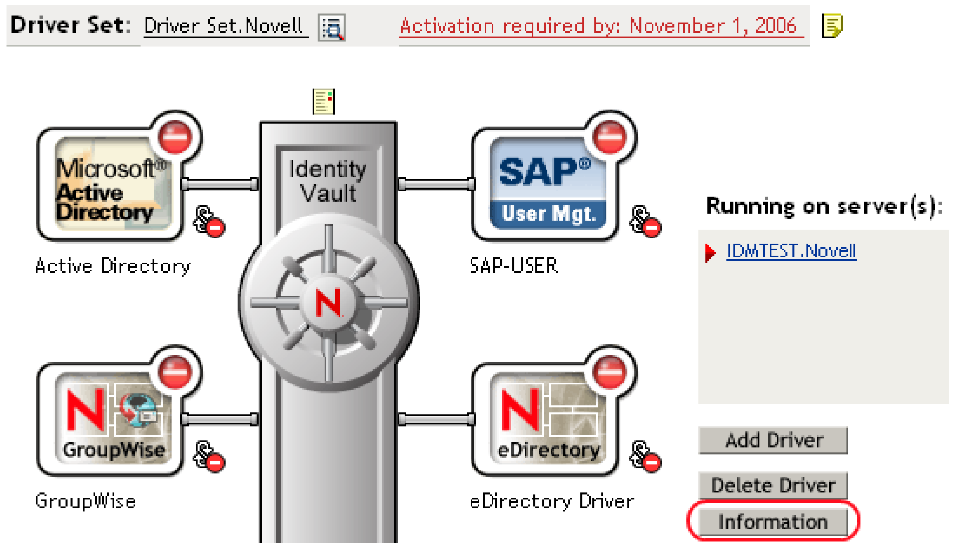
- The driver ID
- The version of the instance of the driver running on that server

7.4.2 Viewing the Version Information As a Text File

Identity Manager publishes version information to a file. You can view this information in text format. The textual representation is the same information contained in the hierarchical view.

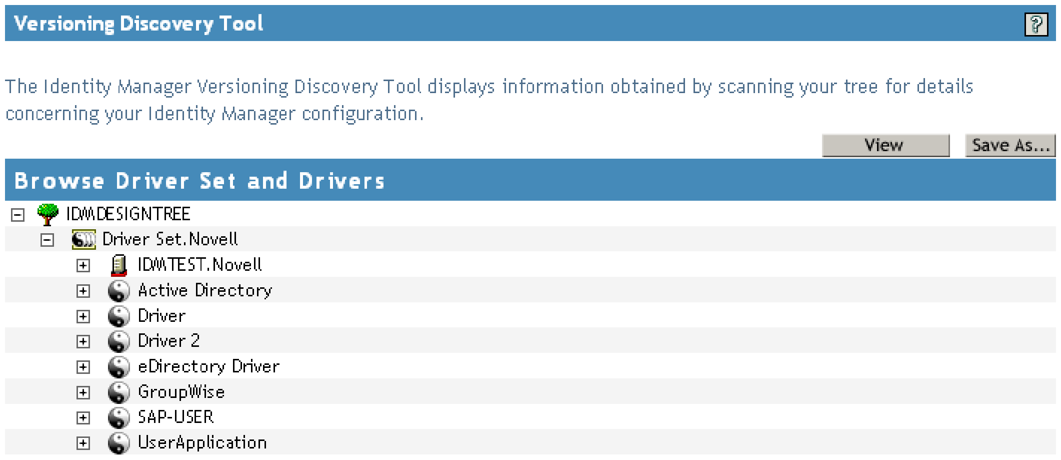
1 To find your Driver Set object in iManager, click *Identity Manager* > *Identity Manager Overview*, then click *Search*.

2 In the Identity Manager Overview, click *Information*.



You can also select *Identity Manager Utilities > Versions Discovery*, browse to and select the Driver Set object, then click *Information*.

3 In the Versioning Discovery Tool dialog box, click *View*.



Versioning Discovery Tool

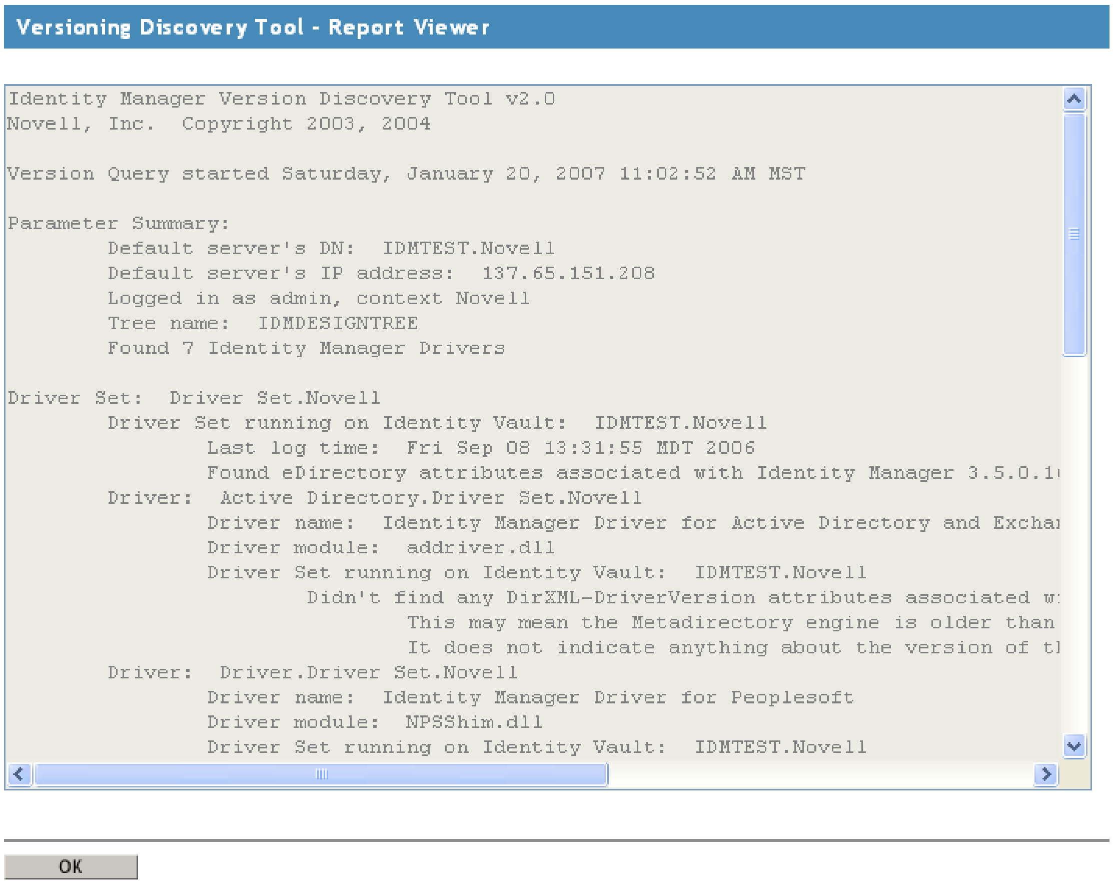
The Identity Manager Versioning Discovery Tool displays information obtained by scanning your tree for details concerning your Identity Manager configuration.

View Save As...

Browse Driver Set and Drivers

- IDMDESIGNTREE
  - Driver Set.Novell
    - IDMTEST.Novell
    - Active Directory
    - Driver
    - Driver 2
    - eDirectory Driver
    - GroupWise
    - SAP-USER
    - UserApplication

The information is displayed as a text file in the Report Viewer window.

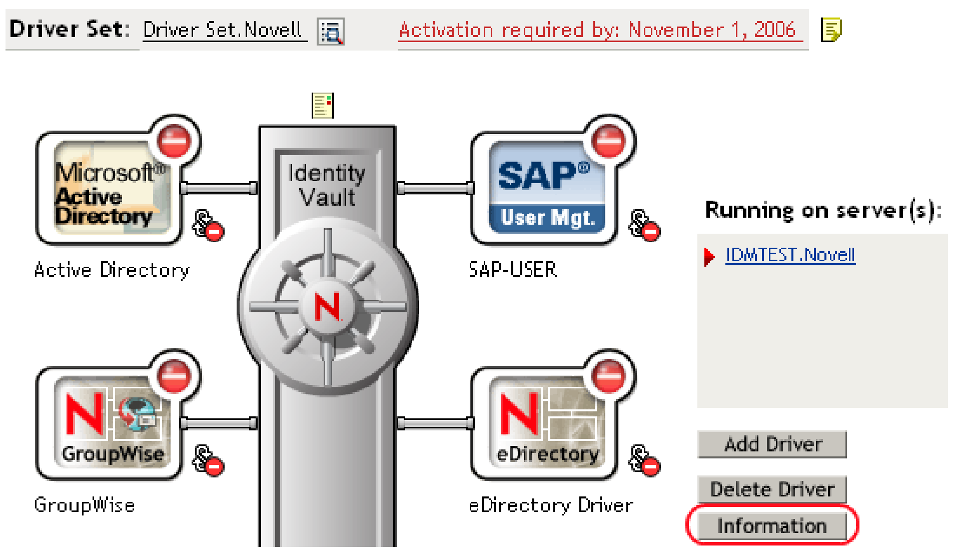


```
Identity Manager Version Discovery Tool v2.0
Novell, Inc.  Copyright 2003, 2004

Version Query started Saturday, January 20, 2007 11:02:52 AM MST

Parameter Summary:
        Default server's DN:  IDMTEST.Novell
        Default server's IP address:  137.65.151.208
        Logged in as admin, context Novell
        Tree name:  IDMDESIGNTREE
        Found 7 Identity Manager Drivers

Driver Set:  Driver Set.Novell
        Driver Set running on Identity Vault:  IDMTEST.Novell
                Last log time:  Fri Sep 08 13:31:55 MDT 2006
                Found eDirectory attributes associated with Identity Manager 3.5.0.1
        Driver:  Active Directory.Driver Set.Novell
                Driver name:  Identity Manager Driver for Active Directory and Excha
                Driver module:  addriver.dll
                Driver Set running on Identity Vault:  IDMTEST.Novell
                        Didn't find any DirXML-DriverVersion attributes associated w
                                This may mean the Metadirectory engine is older than
                                It does not indicate anything about the version of t
        Driver:  Driver.Driver Set.Novell
                Driver name:  Identity Manager Driver for Peoplesoft
                Driver module:  NPSShim.dll
                Driver Set running on Identity Vault:  IDMTEST.Novell
```

7.4.3 Saving Version Information

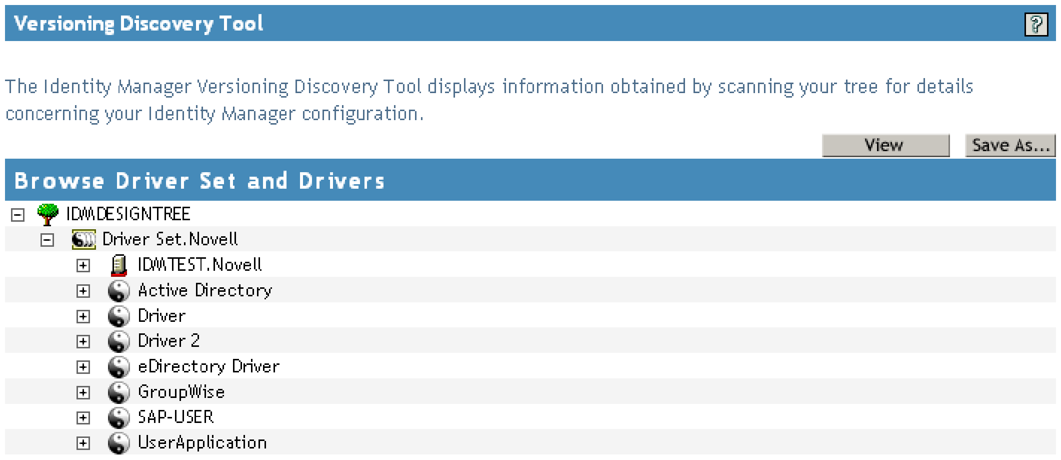
You can save version information to a text file on your local or network drive.

1. To find the Driver Set object in iManager, click *Identity Manager* > *Identity Manager Overview*, then click *Search*.
2. In the Identity Manager Overview, click *Information*.



You can also select *Identity Manager Utilities > Versions Discovery*, browse to and select the Driver Set object, then click *Information*.

3 In the Versioning Discovery Tool dialog box, click *Save As*.


Versioning Discovery Tool

The Identity Manager Versioning Discovery Tool displays information obtained by scanning your tree for details concerning your Identity Manager configuration.

4 In the File Download dialog box, click *Save*.

5 Navigate to the desired directory, type a filename, then click *Save*.

Identity Manager saves the data to a text file.

7.5 Reassociating a Driver Set Object with a Server Object

The driver set object should always be associated with a server object. If the driver set is not associated with a server object, none of the drivers in the driver set can start.

If the link between the driver set object and the server object becomes invalid, you see one of the following conditions:

- When upgrading eDirectory on your Identity Manager server, you get the error UniqueSPIException error -783.
- No server is listed next to the driver set in the Identity Manager Overview window.
- A server is listed next to the driver set in the Identity Manager Overview window, but the name is garbled text.

To resolve this issue, disassociate the driver set object and the server object, then reassociate them.

1. In iManager click *Identity Manager > Identity Manager Overview*, then click *Search* to find the driver set object that the driver should be associated with.
2. Click the *Remove server* icon, then click *OK*.
3. Click the *Add server* icon, then browse to and select the server object.
4. Click *OK*.

7.6 Changing the Driver Configuration

If you need to change the driver configuration, Identity Manager allows you to make the change through Designer or iManager.

To change the driver configuration in Designer:

1. Open a project in the Modeler.
2. Right-click the driver line, then select *Properties*.

To change the driver configuration in iManager:

1. Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
2. Browse to the driver, then click the upper right corner of the driver icon.
3. Click *Edit Properties*.

For a listing of all of the configuration fields, see Appendix B, “Properties of the Delimited Text Driver,” on page 91.

7.7 Storing Driver Passwords Securely with Named Passwords

Identity Manager allows you to store multiple passwords securely for a particular driver. This functionality is referred to as Named Passwords. Each different password is accessed by a key, or name.

You can also use the Named Passwords feature to store other pieces of information securely, such as a user name.

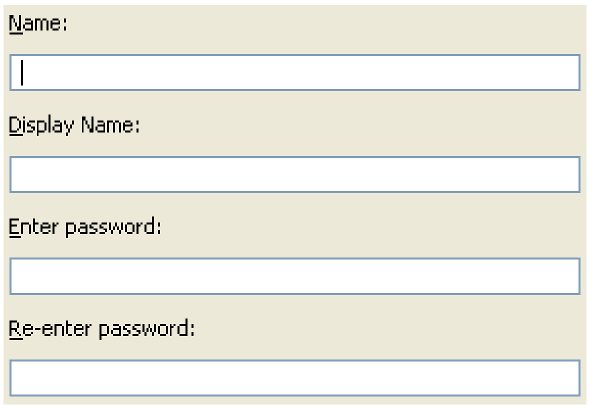
To use a named password in a driver policy, you refer to it by the name of the password, instead of using the actual password, and the Metadirectory engine sends the password to the driver. The method described in this section for storing and retrieving named passwords can be used with any driver without making changes to the driver shim.

- Section 7.7.1, "Using Designer to Configure Named Passwords," on page 49
- Section 7.7.2, "Using iManager to Configure Named Passwords," on page 49
- Section 7.7.3, "Using Named Passwords in Driver Policies," on page 50
- Section 7.7.4, "Configuring Named Passwords by Using the DirXML Command Line Utility," on page 51

7.7.1 Using Designer to Configure Named Passwords

1 Right-click the Driver object, then select *Properties*.

2 Select *Named Password*, then click *New*.



3 Specify the *Name* of the named password.

4 Specify the *Display name* of the named password.

5 Specify the named password, then re-enter the password.

6 Click *OK* twice.

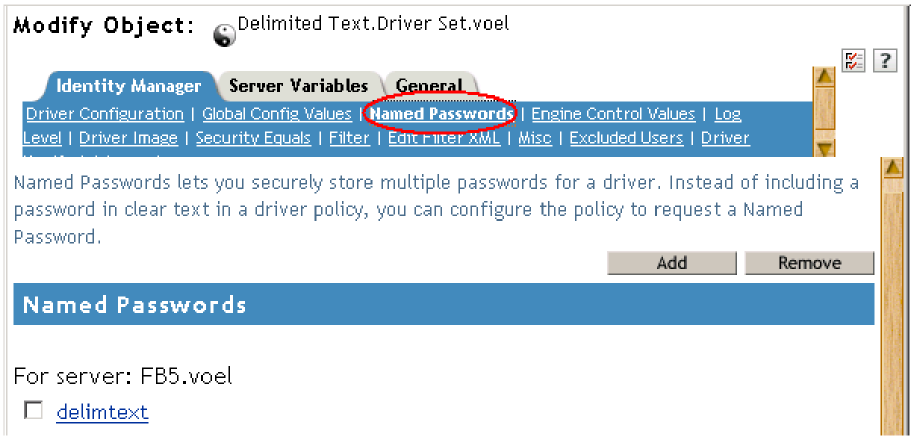
7.7.2 Using iManager to Configure Named Passwords

1 Click *Identity Manager* > *Identity Manager Overview*.

2 Click *Search* to search for the driver set that is associated with the driver.

3 In the Identity Manager Overview, click the upper right corner of the driver icon, then click *Edit properties*.

4 On the Modify Object page, click *Named Passwords*.


Modify Object: Delimited Text.Driver Set.voel

Identity Manager
Server Variables
General
Driver Configuration | Global Config Values | Named Passwords | Engine Control Values | Log Level | Driver Image | Security Equals | Filter | Edit Filter XML | Misc | Excluded Users | Driver

Named Passwords lets you securely store multiple passwords for a driver. Instead of including a password in clear text in a driver policy, you can configure the policy to request a Named Password.

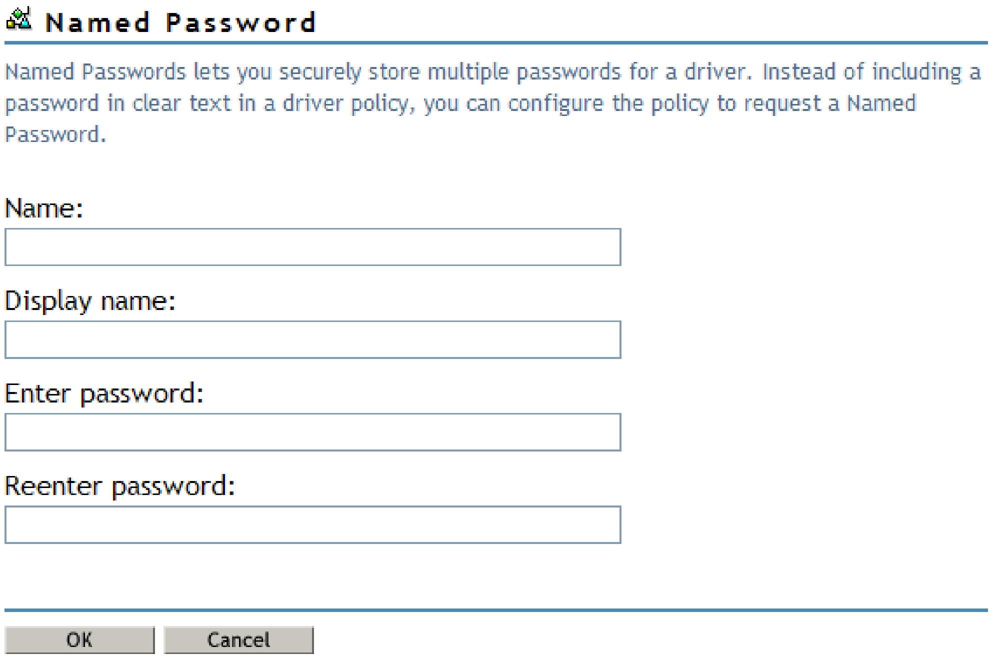
Named Passwords

For server: FB5.voel

☐ delimtext

The Named Passwords page lists the current named passwords for this driver. If you have not set up any named passwords, the list is empty.

5 To add a named password, click *Add*, complete the fields, then click *OK*.


Named Password

Named Passwords lets you securely store multiple passwords for a driver. Instead of including a password in clear text in a driver policy, you can configure the policy to request a Named Password.

Name:

Display name:

Enter password:

Reenter password:

6 Specify a name, display name, and a password, then click *OK* twice.

You can use this feature to store other kinds of information securely, such as a username.

7 Click *OK* to restart the driver and have the changes take effect.

To remove a Named Password, select the password name, then click *Remove*. The password is removed without prompting you to confirm the action.

7.7.3 Using Named Passwords in Driver Policies

- "Making a Call to a Named Password" on page 51
- "Referencing a Named Password" on page 51

Making a Call to a Named Password

Policy Builder allows you to make a call to a named password. Create a new rule and select Named Password as the condition, then set an action, depending upon if the Named Password is available or not available.

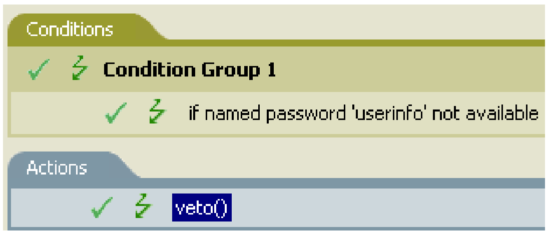
1. In Designer, launch Policy Builder, right-click, then click *New* > *Rule*.
2. Specify the name of the rule, then click *Next*.
3. Select the condition structure, then click *Next*.
4. Select *named password* for the *Condition*.
5. Browse to and select the named password that is stored on the driver.

   In this example, the named password is *userinfo*.
6. Select whether the Operator is available or not available.
7. Select an action for the *Do* field.

   In this example, the action is *veto*.

The example indicates that if the userinfo named password is not available, then the event is vetoed.

Figure 7-1 *A Policy Using Named Passwords*



if named password 'userinfo' not available

veto()

Referencing a Named Password

The following example shows how a named password can be referenced in a driver policy on the Subscriber channel in XSLT:

```
<xsl:value-of
select="query:getNamedPassword($srcQueryProcessor,'mynamedpassword')"
xmlns:query="http://www.novell.com/java/
com.novell.nds.dirxml.driver.XdsQueryProcessor/>
```

7.7.4 Configuring Named Passwords by Using the DirXML Command Line Utility

- "Creating a Named Password in the DirXML Command Line Utility" on page 51
- "Removing a Named Password by Using the DirXML Command Line Utility" on page 53

Creating a Named Password in the DirXML Command Line Utility

1. Run the DirXML Command Line utility.

   For information, see Appendix A, "The DirXML Command Line Utility," on page 77.
2. Enter your username and password.

The following list of options appears.

```
DirXML commands
 1: Start driver
 2: Stop driver
 3: Driver operations...
 4: Driver set operations...
 5: Log events operations...
 6: Get DirXML version
 7: Job operations...
99: Quit
Enter choice:
```

3. Enter `3` for driver operations.

   A numbered list of drivers appears.

4. Enter the number for the driver you want to add a named password to.

   The following list of options appears.

```
Select a driver operation for:
driver_name
 1: Start driver
 2: Stop driver
 3: Get driver state
 4: Get driver start option
 5: Set driver start option
 6: Resync driver
 7: Migrate from application into DirXML
 8: Submit XDS command document to driver
 9: Submit XDS event document to driver
10: Queue event for driver
11: Check object password
12: Initialize new driver object
13: Passwords operations
14: Cache operations
99: Exit
Enter choice:
```

5. Enter `13` for password operations.

   The following list of options appears.

```
Select a password operation
 1: Set shim password
 2: Reset shim password
 3: Set Remote Loader password
 4: Clear Remote Loader password
 5: Set named password
 6: Clear named password(s)
 7: List named passwords
 8: Get passwords state
99: Exit
Enter choice:
```

6. Enter `5` to set a new named password.

   The following prompt appears:

```
Enter password name:
```

7. Enter the name by which you want to refer to the named password.
8. Enter the actual password that you want to secure at the following prompt:

   ```
   Enter password:
   ```

   The characters you type for the password are not displayed.
9. Confirm the password by entering it again at the following prompt:

   ```
   Confirm password:
   ```

10. After you enter and confirm the password, you are returned to the password operations menu.
11. After completing this procedure, use the 99 option twice to exit the menu and quit the DirXML Command Line Utility.

Removing a Named Password by Using the DirXML Command Line Utility

This option is useful if you no longer need named passwords that you previously created.

1. Run the DirXML Command Line utility.

   For information, see Appendix A, "The DirXML Command Line Utility," on page 77.
2. Enter your username and password.

   The following list of options appears.

   ```
   DirXML commands
    1: Start driver
    2: Stop driver
    3: Driver operations...
    4: Driver set operations...
    5: Log events operations...
    6: Get DirXML version
    7: Job operations
   99: Quit
   Enter choice:
   ```

3. Enter 3 for driver operations.

   A numbered list of drivers appears.
4. Enter the number for the driver you want to remove named passwords from.

   The following list of options appears.

   ```
   Select a driver operation for:
   driver_name
    1: Start driver
    2: Stop driver
    3: Get driver state
    4: Get driver start option
    5: Set driver start option
    6: Resync driver
    7: Migrate from application into DirXML
    8: Submit XDS command document to driver
    9: Submit XDS event document to driver
   10: Queue event for driver
   11: Check object password
   ```

```
12: Initialize new driver object
13: Passwords operations
14: Cache operations
99: Exit
Enter choice:
```

5. Enter `13` for password operations.

   The following list of options appears.

```
Select a password operation
 1: Set shim password
 2: Reset shim password
 3: Set Remote Loader password
 4: Clear Remote Loader passwor
 5: Set named password
 6: Clear named password(s)
 7: List named passwords
 8: Get passwords state
99: Exit
Enter choice:
```

6. (Optional) Enter `7` to see the list of existing named passwords.

   The list of existing named passwords is displayed.

   This step can help you make sure you are removing the correct password.
7. Enter `6` to remove one or more named passwords.
8. Enter `No` to remove a single named password at the following prompt:

```
Do you want to clear all named passwords? (yes/no):
```

9. Enter the name of the named password you want to remove at the following prompt:

```
Enter password name:
```

   After you enter the name of the named password you want to remove, you are returned to the password operations menu:

```
Select a password operation
 1: Set shim password
 2: Reset shim password
 3: Set Remote Loader password
 4: Clear Remote Loader password
 5: Set named password
 6: Clear named password(s)
 7: List named passwords
 8: Get passwords state
99: Exit
Enter choice:
```

10. (Optional) Enter `7` to see the list of existing named passwords.

    This step lets you verify that you have removed the correct password.
11. After completing this procedure, use the 99 option twice to exit the menu and quit the DirXML Command Line utility.

7.8 Adding a Driver Heartbeat

The driver heartbeat is a feature of the Identity Manager drivers that ship with Identity Manager 2 and later. Its use is optional. The driver heartbeat is configured by using a driver parameter with a time interval specified. If a heartbeat parameter exists and has an interval value other than 0, the driver sends a heartbeat document to the Metadirectory engine if no communication occurs on the Publisher channel for the specified interval of time.

The intent of the driver heartbeat is to give you a trigger to allow you to initiate an action at regular intervals, if the driver does not communicate on the Publisher channel as often as you want the action to occur. To take advantage of the heartbeat, you must customize your driver configuration or other tools. The Metadirectory engine accepts the heartbeat document but does not take any action because of it.

For most drivers, a driver parameter for heartbeat is not used in the example configurations, but you can add it.

A custom driver that is not provided with Identity Manager can also provide a heartbeat document, if the driver developer has written the driver to support it.

To configure the heartbeat:

1. In iManager, click *Identity Manager* > *Identity Manager Overview*.
2. Browse to and select your driver set object, then click *Search*.
3. In the Identity Manager Overview, click the upper right corner of the driver icon, then click *Edit properties*.
4. On the Identity Manager tab, click *Driver Configuration*, scroll to *Driver Parameters*, then look for Heart Beat or a similar display name.

   If a driver parameter already exists for heartbeat, you can change the interval and save the changes. Configuration is then complete.

   The value of the interval cannot be less than 1. A value of 0 means that the feature is turned off.

   The unit of time is usually minutes; however, some drivers might choose to implement it differently, such as using seconds.
5. If a driver parameter does not exist for heartbeat, click *Edit XML*.
6. Add a driver parameter entry similar to the following example, as a child of `<publisher-options>`.

   ```
   <pub-heartbeat-interval display-name="Heart Beat">10</pub-
   heartbeat-interval>
   ```

   TIP: If the driver does not produce a heartbeat document after being restarted, check the placement of the driver parameter in the XML.
7. Save the changes, then make sure the driver is stopped and restarted.

After you have added the driver parameter, you can use the grapphical view to edit the time interval. Another option is to create a reference to a global configuration value (GCV) for the time interval. Like other global configuration values, the driver heartbeat can be set at the driver set level instead of on each individual Driver object. If a driver does not have a particular global configuration value, and the Driver Set object does have it, the driver inherits the value from the Driver Set object.

Synchronizing Objects 8

This section explains driver and object synchronization in DirXML® 1.1a, Identity Manager 2.0, and Identity Manager 3.*x*. Driver synchronization was not available for DirXML 1.0 and DirXML 1.1.

After the driver is created, instead of waiting for objects to be modified or created, the data between the two connected systems can be sent through the synchronization process.

- Section 8.1, "What Is Synchronization?," on page 57
- Section 8.2, "When Does Synchronization Occur?," on page 57
- Section 8.3, "How Does the Metadirectory Engine Decide Which Object to Synchronize?," on page 58
- Section 8.4, "How Synchronization Works," on page 59

8.1 What Is Synchronization?

The actions commonly referred to as "synchronization" in Identity Manager refer to several different but related actions:

- Synchronization (or merging) of attribute values of an object in the Identity Vault with the corresponding attribute values of an associated object in a connected system.
- Migration of all Identity Vault objects and classes that are included in the filter on the Subscriber channel.
- Generation of the list of objects to submit to the driver's Subscriber channel for synchronization or migration in response to a user request (a manual synchronization).
- Generation of the list of objects to submit to the driver's Subscriber channel for synchronization or migration in response to enabling a formerly disabled driver, or in response to a cache error.

8.2 When Does Synchronization Occur?

The Metadirectory engine synchronizes objects or merges them in the following circumstances:

- When a `<sync>` event element is submitted on the Subscriber or Publisher channel.
- When a `<sync>` event element is submitted on the Subscriber channel in the following circumstances:
  - The state of the object's association value is set to "manual" or "migrate." (This causes an eDirectory™ event, which in turn causes the Identity Manager caching system to queue an object synchronization command in the affected driver's cache.)
  - An object synchronization command is read from the driver's cache.
- When a `<sync>` event element is submitted on the Publisher channel in the following circumstances:
  - A driver submits a `<sync>` event element. No known driver currently does this.

    - The Metadirectory engine submits a `<sync>` event element for each object found as the result of a migrate-into-NDS query. The engine submits these `<sync>` events by using the Subscriber thread, but processes them by using the Publisher channel filter and policies.
- When an `<add>` event (real or synthetic) is submitted on a channel, and the channel Matching policy finds a matching object in the target system.
- When an `<add>` event with an association is submitted on the Subscriber channel. This normally occurs only in exceptional cases, such as the bulk load of objects into eDirectory with DirXML-Associations attribute values.
- When an `<add>` event is submitted on the Publisher channel, and an object is found in eDirectory that already has the association value reported with the `<add>` event.

The Metadirectory engine generates synchronization requests for zero or more objects in the following cases:

- The user issues a manual driver synchronization request. This corresponds to the *Resync* button in the Driver Set property page in ConsoleOne®, or to the *Synchronize* button on the iManager Identity Manager Driver Overview page.
- The Metadirectory engine encounters an error with the driver's cache and cannot recover from the cache error. The driver's cache is deleted, and the engine generates object synchronization commands as detailed in "How Does the Metadirectory Engine Decide Which Object to Synchronize?" on page 58.

8.3 How Does the Metadirectory Engine Decide Which Object to Synchronize?

The Metadirectory engine processes both manually initiated and automatically initiated synchronization requests in the same manner. The only difference in the processing of manually initiated versus automatically initiated driver synchronization requests is the starting filter time used to filter objects being considered for synchronization.

The starting filter time is used to filter objects that have modification or creation times that are older than the starting time specified in the synchronization request.

For automatically initiated driver synchronization, the starting filter time is obtained from the time stamps of cached eDirectory™ events. In particular, the starting filter time is the earliest time for the cached events that haven't yet been successfully processed by the driver's Subscriber channel.

For manually initiated driver synchronization, the default starting filter time is the earliest time in the eDirectory database. In Identity Manager 2 and Identity Manager 3, an explicit starting filter time can also be set. DirXML 1.1a has no facility to set the starting filter time value for synchronization when manually initiating driver synchronization.

The Metadirectory engine creates a list of objects to be synchronized on the Subscriber channel in the following manner:

1. It finds all objects that:
    - Have an entry modification time stamp greater than or equal to the starting filter time

      and
    - Exist in the filter on the Subscriber channel.

2. It finds all objects that have an entry creation time stamp greater than or equal to the starting filter time.
3. It adds a `synchronize object` command to the following:
    - The driver cache for each unique object found that has an entry modification time stamp greater than or equal to the starting filter time
    - All objects and classes that are in the Subscriber filter channel in the driver being synchronized

8.4 How Synchronization Works

After the Metadirectory engine determines that an object is to be synchronized, the following processes occur:

1. Each system (the Identity Vault and the connected system) is queried for all attribute values in the appropriate filters.
    - eDirectory is queried for all values in the Subscriber filter, and for values that are marked for synchronization in Identity Manager 2.*x* and Identity Manager 3.*x*.
    - The connected system is queried for all values in the Publisher filter, and for values that are marked for synchronization in Identity Manager 2.*x* and Identity Manager 3.*x*.
2. The returned attribute values are compared, and modification lists are prepared for the Identity Vault and the connected system according to Table 8-1 on page 60, Table 8-2 on page 62, and Table 8-3 on page 63.

    In the tables the following pseudo-equations are used:
    - "Left = Right" indicates that the left side receives all values from the right side.
    - "Left = Right[1]" indicates that the left side receives one value from the right side. If there is more than one value, it is indeterminate.
    - "Left += Right" indicates that the left side adds the right side values to the left side's existing values.
    - "Left = Left + Right" indicates that the left sides receives the union of the values of the left and right sides.

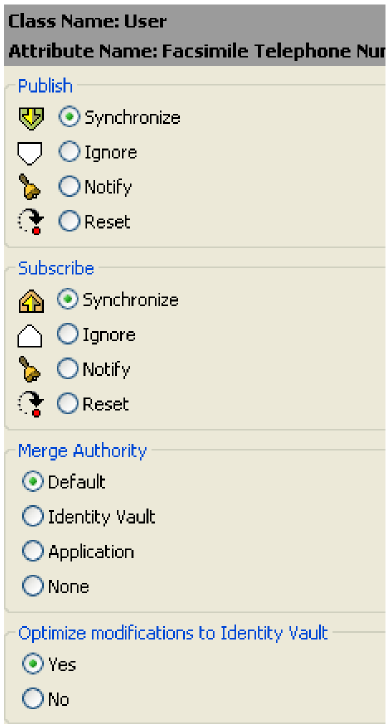
Identity Manager has three different combinations of selected items in the filter, and each one creates a different output.

- Section 8.4.1, "Scenario One," on page 59
- Section 8.4.2, "Scenario Two," on page 61
- Section 8.4.3, "Scenario Three," on page 62

8.4.1 Scenario One

The attribute is set to *Synchronize* on the Publisher and Subscriber channels, and the merge authority is set to *Default*.

Figure 8-1 *Scenario One*


Class Name: User
Attribute Name: Facsimile Telephone Nur

Publish
[✓ Synchronize] Ignore Notify Reset

Subscribe
[✓ Synchronize] Ignore Notify Reset

Merge Authority
[✓ Default] Identity Vault Application None

Optimize modifications to Identity Vault
[✓ Yes] No

Table 8-1 on page 60 contains the values that the Metadirectory engine synchronizes when the attribute is sent through a filter that is set to the configuration for Scenario One. The table shows different outputs depending upon the following:

- Whether the attribute comes from the Identity Vault or the Application
- If the attribute is single-valued or multi-valued, and if the attribute is empty or non-empty
- If the attribute is empty or non-empty

Table 8-1 *Output of Scenario One*

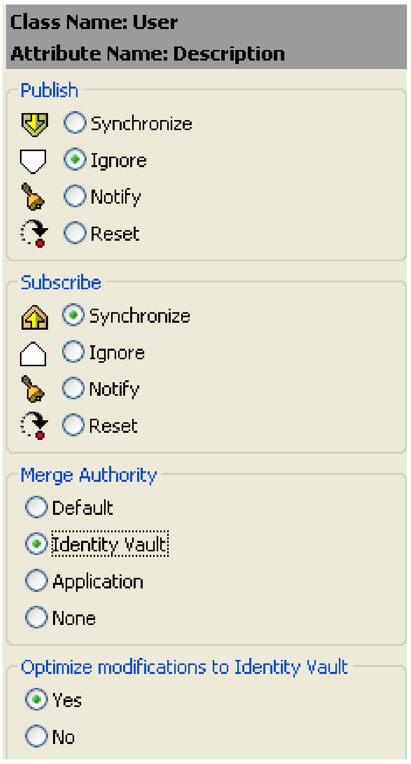
	Identity Vault single-valued empty	**Identity Vault single-valued non-empty**	**Identity Vault multi-valued empty**	**Identity Vault multi-valued non-empty**
Application single-valued empty	No change	App = Identity Vault	No change	App = Identity Vault[1]
Application single-valued non-empty	Identity Vault = App	App = Identity Vault	Identity Vault = App	Identity Vault + = App
Application multi-valued empty	No change	App = Identity Vault	No change	App = Identity Vault

	Identity Vault single-valued empty	Identity Vault single-valued non-empty	Identity Vault multi-valued empty	Identity Vault multi-valued non-empty
Application multi-valued non-empty	Identity Vault = App[1]	App + = Identity Vault	Identity Vault = App	App = App + Identity Vault Identity Vault = App + Identity Vault

8.4.2 Scenario Two

The attribute is set to *Synchronize* only on the Subscriber channel, or it is set to *Synchronize* on both the Subscriber and Publisher channels. The merge authority is set to *Identity Vault*.

Figure 8-2 *Scenario Two*



Class Name: User
Attribute Name: Description

Publish
Synchronize [✓ Ignore] Notify Reset

Subscribe
[✓ Synchronize] Ignore Notify Reset

Merge Authority
Default [✓ Identity Vault] Application None

Optimize modifications to Identity Vault
[✓ Yes] No

Table 8-2 on page 62 contains the values that the Metadirectory engine synchronizes when the attribute is sent through a filter that is set to the configuration for Scenario Two. The table shows different outputs depending upon the following:

- Whether the attribute comes from the Identity Vault or the Application
- If the attribute is single-valued or multi-valued
- If the attribute is empty or non-empty

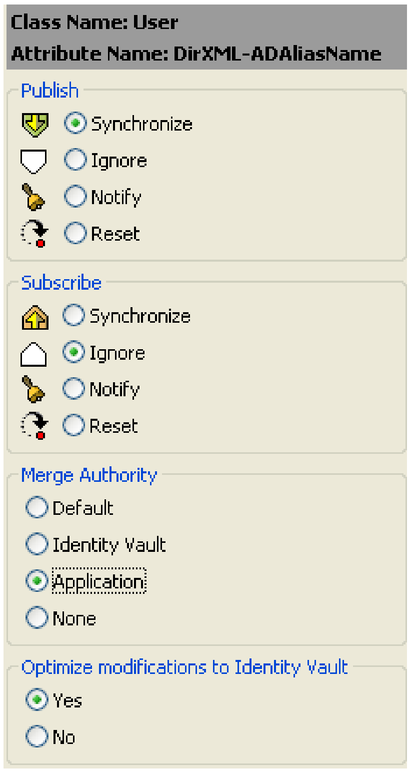
Table 8-2 *Output of Scenario Two*

	Identity Vault single-valued empty	**Identity Vault single-valued non-empty**	**Identity Vault multi-valued empty**	**Identity Vault multi-valued non-empty**
Application single-valued empty	No change	App = Identity Vault	No change	App = Identity Vault[1]
Application single-valued empty	App = empty	App = Identity Vault	Identity Vault = App	App = Identity Vault[1]
Application multi-valued empty	No change	App = Identity Vault	No change	App = Identity Vault
Application multi-valued non-empty	App = empty	App = Identity Vault	App = empty	App = Identity Vault

8.4.3 Scenario Three

The attribute is set to *Synchronize* on the Publisher channel, or the merge authority is set to *Application*.

Figure 8-3 *Scenario Three*



Class Name: User
Attribute Name: DirXML-ADAliasName

Publish
[✓ Synchronize] Ignore Notify Reset

Subscribe
Synchronize [✓ Ignore] Notify Reset

Merge Authority
Default Identity Vault [✓ Application] None

Optimize modifications to Identity Vault
[✓ Yes] No

Table 8-3 on page 63 contains the values that the Metadirectory engine synchronizes when the attribute is sent through a filter that is set to the configuration for Scenario Three. The table shows different outputs depending upon the following:

- Whether the attribute comes from the Identity Vault or the Application
- If the attribute is single-valued or multi-valued
- If the attribute is empty or non-empty

***Table 8-3** Output of Scenario Three*

	Identity Vault single-valued empty	**Identity Vault single-valued non-empty**	**Identity Vault multi-valued empty**	**Identity Vault multi-valued non-empty**
Application single-valued empty	No change	Identity Vault = empty	No change	Identity Vault = empty
Application single-valued non-empty	Identity Vault = App	Identity Vault = App	Identity Vault = App	Identity Vault = App
Application multi-valued empty	No change	Identity Vault = empty	No change	Identity Vault = empty
Application multi-valued non-empty	Identity Vault = App[1]	Identity Vault = App[1]	Identity Vault = App	Identity Vault = App

9 Troubleshooting

Viewing driver processes is necessary to analyze unexpected behavior. To view the driver processing events, use DSTrace. You should only use it during testing and troubleshooting the driver. Running DSTrace while the drivers are in production increases the utilization on the Identity Manager server and can cause events to process very slowly.

- Section 9.1, "Viewing Driver Processes," on page 65

9.1 Viewing Driver Processes

To see the driver processes in DSTrace, values are added to the Driver Set object and the Driver object. You can do this in Designer or iManager.

- Section 9.1.1, "Adding Trace Levels in Designer," on page 65
- Section 9.1.2, "Adding Trace Levels in iManager," on page 67
- Section 9.1.3, "Capturing Driver Processes to a File," on page 67

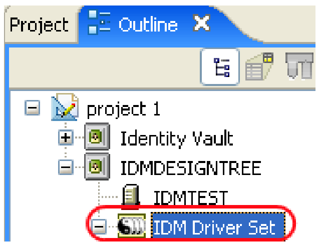
9.1.1 Adding Trace Levels in Designer

You can add trace levels to the Driver Set object or to each Driver object.

- "Driver Set" on page 65
- "Driver" on page 66

Driver Set

1 In an open project in Designer, select the Driver Set object in the *Outline* view.



2 Right-click, select *Properties*, then click *5. Trace*.

3 Set the parameters for tracing, then click *OK*.

Parameter	Description
Driver trace level	As the Driver object trace level increases, the amount of information displayed in DSTrace increases. Trace level 1 shows errors, but not the cause of the errors. To see password synchronization information, set the trace level to 5.

Parameter	Description
XSL trace level	DSTrace displays XSL events. Set this trace level only when troubleshooting XSL style sheets. If you do not want to see XSL information, set the level to zero.
Java debug port	Allows developers to attach a Java debugger.
Java trace file	When a value is set in this field, all Java information for the Driver Set object is written to a file. The value for this field is the path for that file. As long as the file is specified, Java information is written to this file. If you do not need to debug Java, leave this field blank.
Trace file size limit	Allows you to set a limit for the Java trace file. If you set the file size to *Unlimited*, the file grows in size until no disk space remains.

If you set the trace level on the Driver Set object, all drivers appear in the DSTrace logs.

Driver

1 In an open project in Designer, select the Driver object in the *Outline* view.

2 Right-click, select *Properties*, then click *8. Trace*.

3 Set the parameters for tracing, then click *OK*.

Parameter	Description
Trace level	As the Driver object trace level increases, the amount of information displayed in DSTrace increases. Trace level 1 shows errors, but not the cause of the errors. To see password synchronization information, set the trace level to 5. if you select *Use setting from Driver Set*, the value is taken from the Driver Set object.
Trace file	Specify a filename and location for where the Identity Manager information is written for the selected driver. if you select *Use setting from Driver Set*, the value is taken from the Driver Set object.
Trace file size limit	Allows you to set a limit for the Java trace file. If you set the file size to *Unlimited*, the file grows in size until no disk space remains. If you select *Use setting from Driver Set*, the value is taken from the Driver Set object.
Trace name	The driver trace messages are prepended with the value entered instead of the driver name. Use this option if the driver name is very long.

If you set the parameters only on the Driver object, only information for that driver appears in the DSTrace log.

9.1.2 Adding Trace Levels in iManager

You can add trace levels to the Driver Set object or to each Driver object.

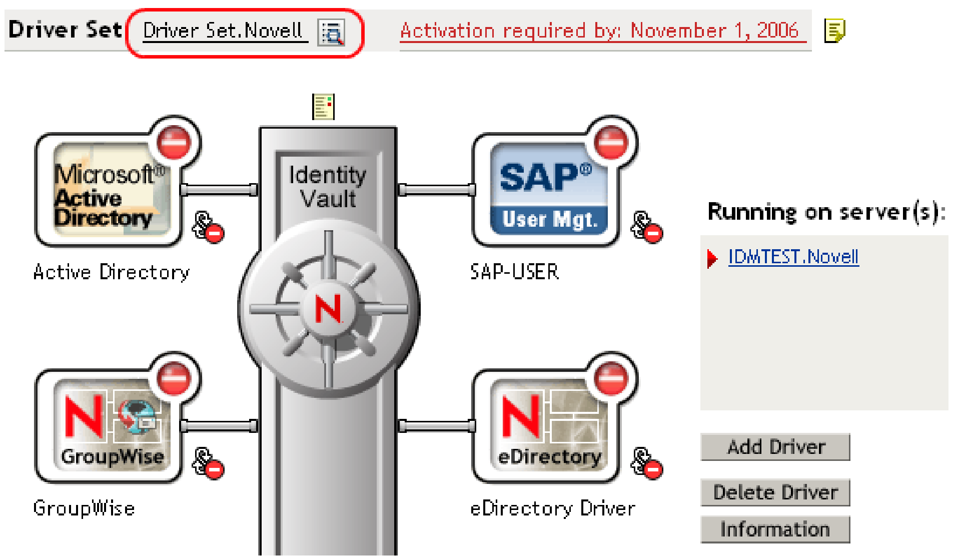
- "Driver Set" on page 67
- "Driver" on page 67

Driver Set

1 In iManager, select *Identity Manager > Identity Manager Overview*.

2 Browse to the Driver Set object, then click *Search*.

3 Click the driver set name.



4 Select the *Misc* tab for the Driver Set object.

5 Set the parameters for tracing, then click *OK*.

Driver

1 In iManager, select *Identity Manager > Identity Manager Overview*.

2 Browse to the Driver Set object where the Driver object resides, then click *Search*.

3 Click the upper right corner of the Driver object, then click *Edit properties*.

4 Select the *Misc* tab for the Driver object.

5 Set the parameters for tracing, then click *OK*.

NOTE: The option *Use setting from Driver Set* does not exist in iManager.

9.1.3 Capturing Driver Processes to a File

You can save driver processes to a file by using the parameter on the Driver object or by using DSTrace. The parameter on the Driver object is the *Trace file* parameter, under the *MISC* tab.

The driver processes that are captured through DSTrace are the processes that occur on the Identity Manager engine. If you use the Remote Loader, you need to capture a trace on the Remote Loader at the same time as you are capturing the trace on the Identity Manager engine.

The following methods help you capture and save Identity Manager processes through DSTrace on different platforms.

- "NetWare" on page 68
- "Windows" on page 68
- "UNIX" on page 69
- "iMonitor" on page 69
- "Remote Loader" on page 70

NetWare

Use `dstrace.nlm` to display trace messages on the system console or trace messages to a file (`sys:\system\dstrace.log`). Use `dstrace.nlm` to display the trace messages to a screen labeled DSTrace Console.

1. Enter `dstrace.nlm` at the server console to load `dstrace.nlm` into memory.
2. Enter `dstrace screen on` at the server console to allow trace messages to appear on the `DSTrace Console` screen.
3. Enter `dstrace file on` at the server console to capture trace messages sent to the DSTrace Console to the `dstrace.log` file.
4. (Optional) Enter `dstrace -all` at the server console to make it easier to read the trace log.
5. Enter `dstrace +dxml dstrace +dvrs` at the server console to display Identity Manager events.
6. Enter `dstrace +tags dstrace +time` at the server console to display message tags and time stamps.
7. Toggle to the DSTrace Console screen and watch for the event to pass.
8. Toggle back to the server console.
9. Enter `dstrace file off` at the server console.

   This stops capturing trace messages to the log file. It also stops logging information into the file.
10. Open the `dstrace.log` in a text editor and search for the event or the object you modified.

Windows

1. Open the Control Panel, select *NDS Services* > `dstrace.dlm`, then click *Start* to display the NDS Server Trace utility window.
2. Click *Edit* > *Options*, then click *Clear All* to clear all of the default flags.
3. Select *DirXML* and *DirXML Drivers*.
4. Click OK.
5. Click *File* > *New*.
6. Specify the filename and location where you want the DSTrace information saved, then click *Open*.

7 Wait for the event to occur.

8 Click *File* > *Close*.

This stops the information from being written to the log file.

9 Open the file in a text editor and search for the event or the object you modified.

UNIX

1 Enter `ndstrace` to start the ndstrace utility.

2 Enter `set ndstrace=nodebug` to turn off all trace flags currently set.

3 Enter `set ndstrace on` to display trace messages to the console.

4 Enter `set ndstrace file on` to capture trace messages to the `ndstrace.log` file in the directory where eDirectory is installed. By default it is `/var/nds`.

5 Enter `set ndstrace=+dxml` to display the Identity Manager events.

6 Enter `set ndstrace=+dvrs` to display the Identity Manager driver events.

7 Wait for the event to occur.

8 Enter `set ndstrace file off` to stop logging information to the file.

9 Enter `exit` to quite the ndstrace utility.

10 Open the file in a text editor. Search for the event or the object that was modified.

iMonitor

iMonitor allows you to get DSTrace information from a Web browser. It does not matter where Identity Manager is running. The following files run iMonitor:

- `ndsimon.nlm` runs on NetWare®.
- `ndsimon.dlm` runs on Windows.
- `ndsimonitor` runs on UNIX*.

1 Access iMonitor from http://*server_ip*:8008/nds.

Port 8008 is the default.

2 Specify a username and password with administrative rights, then click *Login*.

3 Select *Trace Configuration* on the left side.

4 Click *Clear All*.

5 Select *DirXML* and *DirXML Drivers*.

6 Click *Trace On*.

7 Select *Trace History* on the left side.

8 Click the document with the *Modification Time* of *Current* to see a live trace.

9 Change the *Refresh Interval* if you want to see information more often.

10 Select *Trace Configuration* on the left side, then click *Trace Off* to turn the tracing off.

11 Select *Trace History* to view the trace history.

The files are distinguished by the time stamp.

If you need a copy of the HTML file, the default location is:

- NetWare: `sys:\system\ndsimon\dstrace*.htm`
- Windows: `Drive_letter:\novell\nds\ndsimon\dstrace\*.htm`
- UNIX: `/var/nds/dstrace/*.htm`

Remote Loader

You can capture the events that occur on the machine running the Remote Loader service.

1. Launch the Remote Loader Console by clicking the icon.
2. Select the driver instance, then click *Edit*.
3. Set the *Trace Level* to 3 or above.
4. Specify a location and file for the trace file.
5. Specify the amount of disk space that the file is allowed.
6. Click *OK* twice to save the changes.

You can also enable tracing from the command line by using the following switches. For more information, see "Deciding Whether to Use the Remote Loader" in the *Novell Identity Manager 3.5.1 Administration Guide*.

Table 9-1 *Command Line Tracing Switches*

Option	Short Name	Parameter	Description
-trace	-t	integer	Specifies the trace level. This is used only when hosting an application shim. Trace levels correspond to those used on the Identity Manager server. Example: `-trace 3` or `-t3`
-tracefile	-tf	filename	Specify a file to write trace messages to. Trace messages are written to the file if the trace level is greater than zero. Trace messages are written to the file even if the trace window is not open. Example: `-tracefile c:\temp\trace.txt` or `-tf c:\temp\trace.txt`

Option	Short Name	Parameter	Description
-tracefilemax	-tfm	size	Specifies the approximate maximum size that trace file data can occupy on disk. If you specify this option, Identity Manager creates a trace file with the name specified by using the tracefile option and up to 9 additional “roll-over” files. The roll-over files are named by using the base of the main trace filename plus “_n”, where n is 1 through 9. The size parameter is the number of bytes. Specify the size by using the suffixes K, M, or G for kilobytes, megabytes, or gigabytes. If the trace file data is larger than the specified maximum when the Remote Loader is started, the trace file data remains larger than the specified maximum until roll-over is completed through all 10 files. Example: `-tracefilemax 1000M` or `-tfm 1000M`

10 Backing Up the eDirectory Driver

You can use Designer for Identity Manager or iManager to create an XML file of the driver. The file contains all of the information that you entered into the driver during configuration. If the driver becomes corrupted, you can restore the configuration information by importing the exported file.

IMPORTANT: If the driver has been deleted, all of the associations on the objects are purged. When you import the XML file, the migration process creates new associations.

Not all server-specific information stored on the driver is contained in the XML file. Make sure that this information is documented through the Document Generation process in Designer. See "Documenting Projects" in the *Designer 2.1 for Identity Manager 3.5.1* guide.

- Section 10.1, "Exporting the Driver in Designer," on page 73
- Section 10.2, "Exporting the Driver in iManager," on page 73

10.1 Exporting the Driver in Designer

1. Open a project in Designer, then right-click the Driver object.
2. Select *Export to Configuration File*.
3. Specify a unique name for the configuration file, browse to location where it should be saved, then click *Save*.
4. Click *OK* in the Export Configuration Results window.

10.2 Exporting the Driver in iManager

1. In iManager, select *Identity Manager > Identity Manager Overview*.
2. Browse to and select the Driver Set object, then click *Search*.
3. Click the driver icon.
4. Select *Export* in the Identity Manager Driver Overview window.
5. Browse to and select the Driver object that you want to export, then click *Next*.
6. Select *Export all policies, linked to the configuration or not* or select *Only export policies that are linked to the configuration*, depending upon the information you want to have stored in the XML file.
7. Click *Next*.
8. Click *Save As*, then click *Save*.
9. Browse to and select a location to save the XML file, then click *Save*.
10. Click *Finish*.

Security: Best Practices

11

To secure the driver and the information it is synchronizing, see "Security: Best Practices" in the *Novell Identity Manager 3.5.1 Administration Guide*.

The DirXML Command Line Utility

A

The DirXML® Command Line utility allows you to use a command line interface to manage the driver. You can create scripts that have the commands to manage the driver.

The utility and scripts are installed on all platforms during the Identity Manager installation. The utility is installed to the following locations:

- Windows: `\Novell\Nds\dxcmd.bat`
- NetWare®: `sys:\system\dxcmd.ncf`
- UNIX: `/usr/bin/dxcmd`

Either of the following methods enable you to use the DirXML Command Line utility:

- Section A.1, "Interactive Mode," on page 77
- Section A.2, "Command Line Mode," on page 86

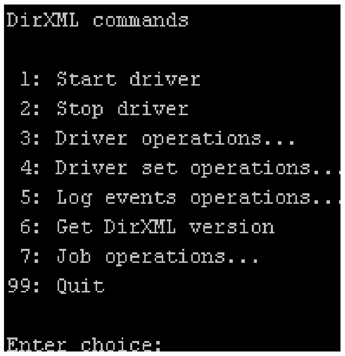
A.1 Interactive Mode

The interactive mode provides a text interface to control and use the DirXML Command Line utility.

1 At the console, enter `dxcmd`.

2 Enter the name of a user with sufficient rights to the Identity Manager objects, such as admin.novell.

3 Enter the user's password.



```
DirXML commands

 1: Start driver
 2: Stop driver
 3: Driver operations...
 4: Driver set operations..
 5: Log events operations..
 6: Get DirXML version
 7: Job operations...
99: Quit

Enter choice:
```

4 Enter the number of the command that you want to perform.

Table A-1 on page 78 contains the list of options and what functionality is available.

5 To quit the utility, enter 99.

NOTE: If you are running eDirectory™ 8.8 on UNIX or Linux", you must specify the -host and -port parameters. For example, `dxcmd -host 10.0.0.1 -port 524`. If the parameters are not specified, a jclient error occurs.

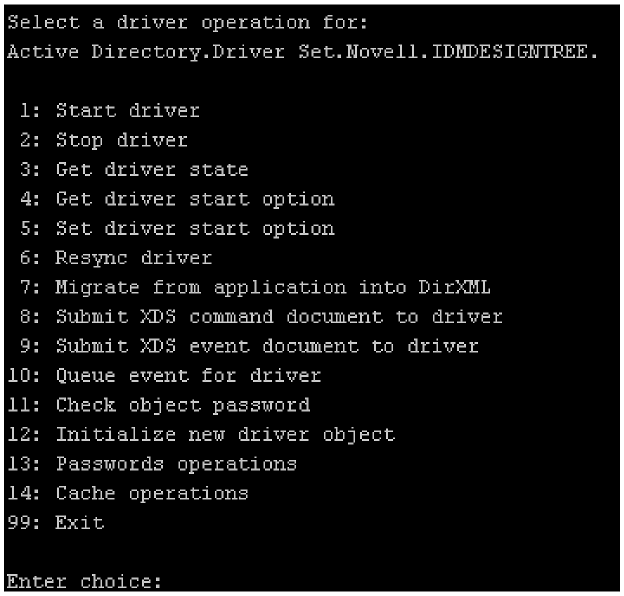
```
novell.jclient.JCException: connect (to address) 111 UNKNOWN ERROR
```

By default, eDirectory 8.8 is not listening to localhost. The DirXML Command Line utility needs to resolve the server IP address or hostname and the port to be able to authenticate.

Table A-1 *Interactive Mode Options*

Option	Description
1: *Start Driver*	Starts the driver. If more than one driver exists, each driver is listed with a number. Enter the number of the driver to start the driver.
2: *Stop Driver*	Stops the driver. If more than one driver exists, each driver is listed with a number. Enter the number of the driver to stop the driver.
3: *Driver operations*	Lists the operations available for the driver. If more than one driver exists, each driver is listed with a number. Enter the number of the driver to see the operations available. See Table A-2 on page 79 for a list of operations.
4: *Driver set operations*	Lists the operations available for the driver set. ◆ 1: Associate driver set with server ◆ 2: Disassociate driver set from server ◆ 99: Exit
5: *Log events operations*	Lists the operations available for logging events through Novell® Audit. See Table A-5 on page 83 for a description of these options.
6: *Get DirXML version*	Lists the installed version of Identity Manager.
7: *Job operations*	Manages jobs created for Identity Manager.
99: *Quit*	Exits the DirXML Command Line utility

Figure A-1 *Driver Options*



```
Select a driver operation for:
Active Directory.Driver Set.Novell.IDMDESIGNTREE.

 1: Start driver
 2: Stop driver
 3: Get driver state
 4: Get driver start option
 5: Set driver start option
 6: Resync driver
 7: Migrate from application into DirXML
 8: Submit XDS command document to driver
 9: Submit XDS event document to driver
10: Queue event for driver
11: Check object password
12: Initialize new driver object
13: Passwords operations
14: Cache operations
99: Exit

Enter choice:
```

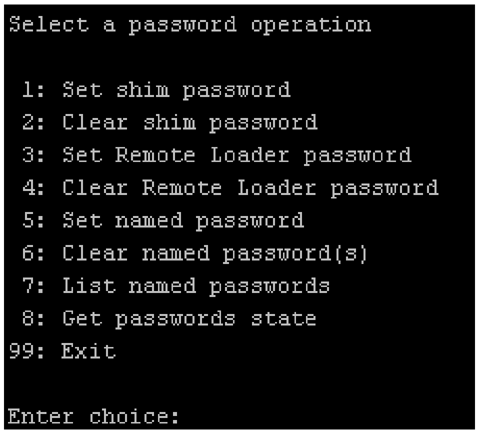
Table A-2 *Driver Options*

Options	Description
1: *Start driver*	Starts the driver.
2: *Stop driver*	Stops the driver.
3: *Get driver state*	Lists the state of the driver. ◆ 0 - Driver is stopped ◆ 1 - Driver is starting ◆ 2 - Driver is running ◆ 3 - Driver is stopping
4: *Get driver start option*	Lists the current driver start option. ◆ 1 - Disabled ◆ 2 - Manual ◆ 3 - Auto
5: *Set driver start option*	Changes the start option of the driver. ◆ 1 - Disabled ◆ 2 - Manual ◆ 3 - Auto ◆ 99 - Exit
6: *Resync driver*	Forces a resynchronization of the driver. It prompts for a time delay: *Do you want to specify a minimum time for resync? (yes/no).* If you enter Yes, specify the date and time you want the resynchronization to occur: *Enter a date/time (format 9/27/05 3:27 PM)*. If you enter No, the resynchronization occurs immediately.
7: *Migrate from application into DirXML*	Processes an XML document that contains a query command: *Enter filename of XDS query document:* Create the XML document that contains a query command by using the Novell `nds.dtd` (http://developer.novell.com/ndk/doc/dirxml/dirxmlbk/ref/ndsdtd/query.html). Examples: NetWare: `sys:\files\query.xml` Windows: `c:\files\query.xml` Linux: `/files/query.xml`

Options	Description
8: *Submit XDS command document to driver*	Processes an XDS command document: *Enter filename of XDS command document:* Examples: NetWare: `sys:\files\user.xml` Windows: `c:\files\user.xml` Linux: `/files/user.xml` *Enter name of file for response:* Examples: NetWare: `sys:\files\user.log` Windows: `c:\files\user.log` Linux: `/files/user.log`
9: Submit XDS event document to driver	Processes an XDS event document: *Enter filename of XDS event document:* Examples: NetWare: `sys:\files\add.xml` Windows: `c:\files\add.xml` Linux: `/files/add.xml`
10: Queue event for driver	Adds an event to the driver queue *Enter filename of XDS event document:* Examples: NetWare: `sys:\files\add.xml` Windows: `c:\files\add.xml` Linux: `/files/add.xml`
11: *Check object password*	Validates that an object's password in the connected system is associated with a driver. It matches the object's eDirectory password (Distribution Password, used with Universal Password). *Enter user name*:
12: *Initialize new driver object*	Performs an internal initialization of data on a new Driver object. This is only for testing purposes.
13: *Password operations*	Nine Password options are available. See Table A-3 on page 81 for a description of these options.
14: *Cache operations*	Five Cache operations exist. See Table A-4 on page 82 for a descriptions of these options.

Options	Description
99: *Exit*	Exits the driver options.

Figure A-2 *Password Operations*



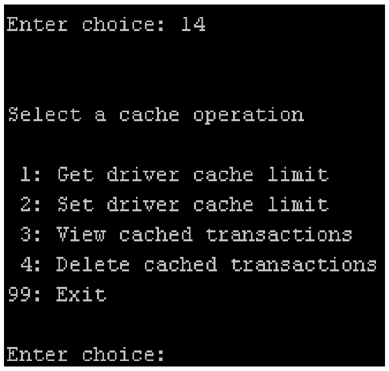
```
Select a password operation

 1: Set shim password
 2: Clear shim password
 3: Set Remote Loader password
 4: Clear Remote Loader password
 5: Set named password
 6: Clear named password(s)
 7: List named passwords
 8: Get passwords state
99: Exit

Enter choice:
```

Table A-3 *Password Operations*

Operation	Description
1: *Set shim password*	Sets the application password. This is the password of the user account you are using to authenticate into the connected system with.
2: *Clear shim password*	Clears the application password.
3: *Set Remote Loader password*	The Remote Loader password is used to control access to the Remote Loader instance. Enter the Remote Loader password, then confirm the password by typing it again.
4: *Clear Remote Loader password*	Clears the Remote Loader password so no Remote Loader password is set on the Driver object.
5: *Set named password*	Allows you to store a password or other pieces of security information on the driver. See Section 7.7, “Storing Driver Passwords Securely with Named Passwords,” on page 48 for more information. Lists four prompts: ◆ *Enter password name:* ◆ *Enter password description:* ◆ *Enter password:* ◆ *Confirm password:*

Operation	Description
6: *Clear named passwords*	Clears a specified named password or all named passwords that are stored on the Driver object: *Do you want to clear all named passwords? (yes/no).* If you enter Yes, all Named Passwords are cleared. If you enter No, you are prompted to specify the password name that you want to clear.
7: *List named passwords*	Lists all named passwords that are stored on the Driver object. It lists the password name and the password description.
8: *Get password state*	Lists if a password is set for: ◆ Driver Object password ◆ Application password ◆ Remote loader password The dxcmd utility enables you to set the Application password and the Remote Loader password. You cannot set the Driver Object password with this utility. It displays whether the password has been set.
99: *Exit*	Exits the current menu and takes you back to the Driver options.

Figure A-3 *Cache Operations*



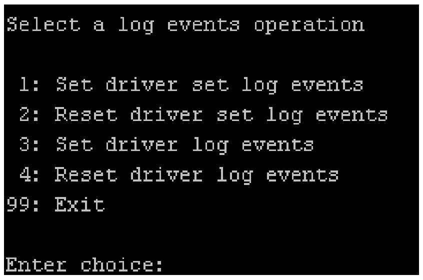
```
Enter choice: 14


Select a cache operation

 1: Get driver cache limit
 2: Set driver cache limit
 3: View cached transactions
 4: Delete cached transactions
99: Exit

Enter choice:
```

Table A-4 *Cache Operations*

Operation	Description
1: *Get driver cache limit*	Displays the current cache limit that is set for the driver.
2: *Set driver cache limit*	Sets the driver cache limit in kilobytes. A value of 0 is unlimited.

Operation	Description
3: *View cached transactions*	A text file is created with the events that are stored in cache. You can select the number of transactions to view. ◆ *Enter option token* (default=0): ◆ *Enter maximum transactions records to return* (default=1): ◆ *Enter name of file for response:*
4: *Delete cached transactions*	Deletes the transactions stored in cache. ◆ *Enter position token* (default=0): ◆ *Enter event-id value of first transaction record* to delete (optional): ◆ *Enter number of transaction records to delete* (default=1):
99: *Exit*	Exits the current menu and takes you back to the Driver options.

Figure A-4 *Log Event Operations*



```
Select a log events operation

 1: Set driver set log events
 2: Reset driver set log events
 3: Set driver log events
 4: Reset driver log events
99: Exit

Enter choice:
```

Table A-5 *Log Events Operations*

Operation	Description
1: *Set driver set log events*	Allows you to log driver set events through Novell Audit. You can select 49 items to log. See Table A-6 on page 84 for a list of these options. Type the number of the item you want to log. After the items are selected, enter 99 to accept the selections.
2: *Reset driver set log events*	Resets all log event options.
3: *Set driver log events*	Allows you to log driver events through Novell Audit. You can select 49 items to log. See Table A-6 on page 84 for a list of these options. Type the number of the item you want to log. After the items are selected, enter 99 to accept the selections.

Operation	Description
4: *Reset driver log events*	Resets all of the log event options.
99: *Exit*	Exits the log events operations menu.

Table A-6 *Driver Set and Driver Log Events*

Options
1: Status success
2: Status retry
3: Status warning
4: Status error
5: Status fatal
6: Status other
7: Query elements
8: Add elements
9: Remove elements
10: Modify elements
11: Rename elements
12: Move elements
13: Add-association elements
14: Remove-association elements
15: Query-schema elements
16: Check-password elements
17: Check-object-password elements
18: Modify-password elements
19: Sync elements
20: Pre-transformed XDS document from shim
21: Post input transformation XDS document
22: Post output transformation XDS document
23: Post event transformation XDS document
24: Post placement transformation XDS document
25: Post create transformation XDS document
26: Post mapping transformation <inbound> XDS document
27: Post mapping transformation <outbound> XDS document

Options
28: Post matching transformation XDS document
29: Post command transformation XDS document
30: Post-filtered XDS document <Publisher>
31: User agent XDS command document
32: Driver resync request
33: Driver migrate from application
34: Driver start
35: Driver stop
36: Password sync
37: Password request
38: Engine error
39: Engine warning
40: Add attribute
41: Clear attribute
42: Add value
43: Remove value
44: Merge entire
45: Get named password
46: Reset Attributes
47: Add Value - Add Entry
48: Set SSO Credential
49: Clear SSO Credential
50: Set SSO Passphrase
51: User defined IDs
99: Accept checked items

Table A-7 *Job Scheduler Operations*

Options	Description
1: *Get available job definitions*	Allows you to select an existing job. *Enter the job number:* *Do you want to filter the job definitions by containment?* Enter Yes or No *Enter name of the file for response:* Examples: NetWare: `sys:\files\user.log` Windows: `c:\files\user.log` Linux: `/files/user.log`
2: *Operations on specific job object*	Allows you to perform operations for a specific job.

A.2 Command Line Mode

The command line mode allows you to use script or batch files. Table A-8 on page 86 lists the different options that are available.

To use the command line options, decide which items you want to use and string them together.

Example: `dxcmd -user admin.headquarters -host 10.0.0.1 -password n0vell -start test.driverset.headquarters`

This example command starts the driver.

Table A-8 *Command Line Options*

Option	Description
Configuration	
-user <*user name*>	Specify the name of a user with administrative rights to the drivers you want to test.
-host <*name or IP address*>	Specify the IP address of the server where the driver is installed.
-password <*user password*>	Specify the password of the user specified above.
-port <*port number*>	Specify a port number, if the default port is not used.
-q <*quiet mode*>	Displays very little information when a command is executed.
-v <*verbose mode*>	Displays detailed information when a command is executed.

Option	Description
-s <*stdout*>	Writes the results of the `dxcmd` command to `stdout`.
-? <*show this message*>	Displays the help menu.
-help <*show this message*>	Displays the help menu.
Actions	
-start <*driver dn*>	Starts the driver.
-stop <*driver dn*>	Stops the driver.
-getstate <*driver dn*>	Shows the state of the driver as running or stopped.
-getstartoption <*driver dn*>	Shows the startup option of the driver.
-setstartoption <*driver dn*> <*disabled*|*manual*|*auto*> <*resync*|*noresync*>	Sets how the driver starts if the server is rebooted. Sets whether the objects are to be resynchronized when the driver restarts.
-getcachelimit <*driver dn*>	Lists the cache limit set for the driver.
-setcachelimit <*driver dn*> <*0 or positive integer*>	Sets the cache limit for the driver.
-migrateapp <*driver dn*> <*filename*>	Processes an XML document that contains a query command. Create the XML document that contains a query command by using the Novell `nds.dtd` (http://www.novell.com/documentation/idm35/index.html?page=/documentation/idm35/policy_dtd/data/dtdndsoverview.html#dtdndsoverview).
-setshimpassword <*driver dn*> <*password*>	Sets the application password. This is the password of the user account you are using to authenticate into the connected system with.
-clearshimpassword <*driver dn*> <*password*>	Clears the application password.
-setremoteloaderpassword <*driver dn*> <*password*>	Sets the Remote Loader password. The Remote Loader password is used to control access to the Remote Loader instance.
<clearremoteloaderpassword <*driver dn*>	Clears the Remote Loader password.

Option	Description
-sendcommand *<driver dn> <input filename> <output filename>*	Processes an XDS command document. Specify the XDS command document as the input file. Examples: NetWare: `sys:\files\user.xml` Windows: `c:\files\user.xml` Linux: `/files/user.log` Specify the output filename to see the results. Examples: NetWare: `sys:\files\user.log` Windows: `c:\files\user.log` Linux: `/files/user.log`
-sendevent *<driver dn> <input filename>*	Submits a document to the driver's Subscriber channel, bypassing the driver cache. The document is processed ahead of anything that might be in the cache at the time of the submission. It also means that the submission fails if the driver is not running.
-queueevent *<driver dn> <input filename>*	Submits a document to the driver's Subscriber channel by queuing the document in the driver cache. The document is processed after anything that might be in the cache at the time of the submission. The submission won't fail if the driver isn't running.
-setlogevents *<dn> <integer ...>*	Sets Novell Audit log events on the driver. The integer is the option of the item to log. See Table A-6 on page 84 for the list of the integers to enter.
-clearlogevents *<dn>*	Clears all Novell Audit log events that are set on the driver.
-setdriverset *<driver set dn>*	Associates a driver set with the server.
-cleardriverset	Clears the driver set association from the server.
-getversion	Shows the version of Identity Manager that is installed.
-initdriver object *<dn>*	Performs an internal initialization of data on a new Driver object. This is only for testing purposes.
-setnamedpassword *<driver dn> <name> <password> [description]*	Sets named passwords on the driver object. You specify the name, the password, and the description of the named password.
-clearnamedpassword *<driver dn> <name>*	Clears a specified named password.
-startjob *<job dn>*	Starts the specified job.

Option	Description
-abortjob *<job dn>*	Aborts the specified job.
-getjobrunningstate *<job dn>*	Returns the specified job's running state.
-getjobenabledstate *<job dn>*	Returns the specified job's enabled state.
-getjobnextruntime *<job dn>*	Returns the specified job's next run time.
-updatejob *<job dn>*	Updates the specified job.
-clearallnamedpaswords *<driver dn>*	Clears all named passwords set on a specific driver.

If a command is executed successfully, it returns a zero. If the command returns anything other than zero, it is an error. For example, 0 means success, and -641 means invalid operation. -641 is an eDirectory error code. Table A-9 on page 89 contains other values for specific command line options.

Table A-9 *Command Line Option Values*

Command Line Option	Values
-getstate	0- stopped
	1- starting
	2- running
	3- shutting down
	11- get schema
	Anything else that is returned is an error.
-getstartoption	0- disabled
	1- manual
	2- auto
	Anything else that is returned is an error.
-getcachelimit	0- unlimited
	Anything else that is returned is an error.
-getjobrunningstate	0- stopped
	1- running
	Anything else that is returned is an error.
-getjobenabledstate	0- disabled
	1- enabled
	2- configuration error
	Anything else that is returned is an error.

Command Line Option	Values
-getjobnextruntime	Returns the next scheduled time for the job in eDirectory time format (number of seconds since 00:00:00 Jan 1, 1970 UTC).

Properties of the Delimited Text Driver

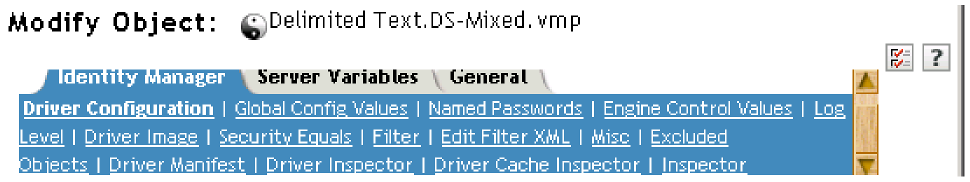
B

This section is a reference for all of the fields on the driver as displayed in iManager and Designer. Sometimes fields are displayed differently in iManager than in Designer.

The information is presented from the viewpoint of iManager. If a field is different in Designer for Identity Manager, it is marked with a Designer icon.

The following figure illustrates property pages in iManager:

Figure B-1 *Delimited Text Driver Properties Pages*



Modify Object: Delimited Text.DS-Mixed.vmp

Identity Manager | Server Variables | General

Driver Configuration | Global Config Values | Named Passwords | Engine Control Values | Log Level | Driver Image | Security Equals | Filter | Edit Filter XML | Misc | Excluded Objects | Driver Manifest | Driver Inspector | Driver Cache Inspector | Inspector

- Section B.1, "Driver Configuration," on page 91
- Section B.2, "Global Configuration Values," on page 99
- Section B.3, "Named Passwords," on page 100
- Section B.4, "Engine Control Values," on page 101
- Section B.5, "Log Level," on page 103
- Section B.6, "Driver Image," on page 104
- Section B.7, "Security Equals," on page 104
- Section B.8, "Filter," on page 104
- Section B.9, "Edit Filter XML," on page 105
- Section B.10, "Misc," on page 105
- Section B.11, "Excluded Users," on page 106
- Section B.12, "Driver Manifest," on page 106
- Section B.13, "Driver Cache Inspector," on page 107
- Section B.14, "Driver Inspector," on page 107
- Section B.15, "Server Variables," on page 108

B.1 Driver Configuration

In iManager:

1. Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
2. Browse to the driver, then click the upper right corner of the driver icon.
3. Click *Edit Properties > Driver Configuration.*

In Designer:

1 Open a project in the Modeler, then right-click the driver line.
2 Click *Properties* > *Driver Configuration.*

To configure the Delimited Text driver, set parameters on the following:

- Section B.1.1, "Driver Module," on page 92
- Section B.1.2, "Module," on page 93
- Section B.1.3, "Driver Object Password," on page 93
- Section B.1.4, "Authentication," on page 94
- Section B.1.5, "Startup Option," on page 95
- Section B.1.6, "Driver Parameters," on page 95
- Section B.1.7, "ECMAScript," on page 99

B.1.1 Driver Module

The driver module changes the driver from running locally to running remotely or the reverse.

In iManager:

1 Click *Identity Manager* > *Identity Manager Overview*.
2 Click *Search* to search for the driver set that is associated with the driver.
3 Click the upper right corner of the driver icon, then click *Edit Properties*.

   By default, iManager displays the I*dentity Manager* tab and the Driver Configuration option. This option has the following sections that you can edit:

   - Module
   - Driver Object Password
   - Authentication
   - Startup Option
   - Driver Parameters

In Designer:

1 Open a project in the Modeler.
2 Right-click the driver line, then select *Properties* > *Driver Configuration.*

   You can edit settings on the following tabs:

   - Driver Module
   - Authentication
   - Startup Option
   - Driver Parameters
   - ECMAScript

B.1.2 Module

***Table B-1** Driver Module Settings*

Option	Description
Java	Used to specify the name of the Java class that is instantiated for the shim component of the driver. This class can be located in the `classes` directory as a class file, or in the `lib` directory as a `.jar` file. If this option is selected, the driver is running locally.
Native	Used to specify the name of the `.dll` file that is instantiated for the application shim component of the driver. If this option is selected, the driver is running locally.
Connect to Remote Loader	Used when the driver is connecting remotely to the connected system.
Remote Loader Client Configuration for Documentation	Includes information on the Remote Loader client configuration when Designer generates documentation for the Delimited Text driver.

B.1.3 Driver Object Password

In iManager:

1. Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
2. Browse to the driver, then click the upper right corner of the driver icon.
3. Click *Edit Properties > Driver Configuration > Driver Object Password > Set Password.*

In Designer:

1. Open a project in the Modeler.
2. Right-click the driver line, then click *Properties > Driver Configuration.*
3. Click *Driver Module > Connect to Remote Loader > Set Password.*

***Table B-2** Driver Object Password*

Option	Description
Driver Object Password	Specifies a password for the Driver object. If you are using the Remote Loader, you must enter a password on this page. Otherwise, the remote driver does not run. The Remote Loader uses this password to authenticate itself to the remote driver shim.

B.1.4 Authentication

The authentication section stores the information required to authenticate to the connected system.

In iManager:

1. Click *Identity Manager > Identity Manager Overview*.
2. Click *Search* to search for the driver set that is associated with the driver.
3. Click the upper right corner of the driver icon, the click *Edit Properties*.
4. Click *Driver Configuration > Authentication*.

In Designer:

1. Open a project in the Modeler.
2. Right-click the driver line, then select *Properties > Driver Configuration*.
3. Click *Authentication*.

Table B-3 *Authentication Settings*

Option	Description
Authentication information for server	Displays or specifies the server that the driver is associated with.
Authentication ID	Specify a user application ID. This ID is used to pass Identity Vault subscription information to the application. Example: `Administrator`
Authentication Context or *Connection Information*	Specify the IP address or name of the server that the application shim should communicate with.
Remote Loader Connection Parameters or *Host name* Port *KMO* *Other parameters*	Used only if the driver is connecting to the application through the Remote Loader. The parameter to enter is `hostname=xxx.xxx.xxx.xxx port=xxxx kmo=certificatename`, when the host name is the IP address of the application server running the Remote Loader server and the port is the port the Remote Loader is listening on. The default port for the Remote Loader is 8090. The `kmo` entry is optional. It is used only when an SSL connection exists between the Remote Loader and the Metadirectory engine. Example: `hostname=10.0.0.1 port=8090 kmo=IDMCertificate`
Driver Cache Limit (kilobytes) or *Cache limit (KB)*	Specify the maximum event cache file size (in KB). If it is set to zero, the file size is unlimited. Click *Unlimited* to set the file size to unlimited in Designer.

Option	Description
Application Password or *Set Password*	Specify the password for the user object listed in the *Authentication ID* field.
Remote Loader Password or *Set Password*	Used only if the driver is connecting to the application through the Remote Loader. The password is used to control access to the Remote Loader instance. It must be the same password specified during the configuration of the Remote Loader on the connected system.

B.1.5 Startup Option

The Startup Option enables you to set the driver state when the Identity Manager server is started.

In iManager:

1 Click *Identity Manager > Identity Manager Overview*.
2 Click *Search* to search for the driver set that is associated with the driver.
3 Click the upper right corner of the driver icon.
4 Click *Edit Properties > Driver Configuration > Startup Option*.

In Designer:

1 Open a project in the Modeler.
2 Right-click the driver line, then select *Properties > Driver Configuration.*
3 Click *Startup Option*.

Table B-4 *Startup Settings*

Option	Description
Auto start	The driver starts every time the Identity Manager server is started.
Manual	The driver does not start when the Identity Manager server is started. The driver must be started through Designer or iManager.
Disabled	The driver has a cache file that stores all of the events. When the driver is set to *Disabled*, this file is deleted and no new events are stored in the file until the driver state is changed to *Manual* or *Auto Start*.
Do not automatically synchronize the driver	This option applies only if the driver is deployed and was previously disabled. If this is not selected, the driver re-synchronizes the next time it is started.

B.1.6 Driver Parameters

In iManager:

1 Click *Identity Manager > Identity Manager Overview*.
2 Click *Search* to search for the driver set that is associated with the driver.

3. Click the upper right corner of the driver icon.
4. Click *Edit Properties > Driver Configuration > Driver Parameters*.

In Designer:

1. Open a project in the Modeler.
2. Right-click the driver line, then select *Properties > Driver Configuration.*
3. Click *Driver Parameters*.

***Table B-5** Driver Parameter Settings*

Parameter	Description
Driver parameters for server	Displays or specifies the server name or IP address of the server whose driver parameters you want to modify.
Edit XML	Opens an editor so that you can edit the driver's configuration file.
Driver Options	
Field Delimiter	Specifies the character that is used to delimit field values in the input files. It must be one character. The default is a comma. If the values of any of the input fields contain this character, enclose the entire value in quotes to prevent it from being seen as a delimiter. Changing this delimiter parameter to something other than a comma does not automatically change the delimiter character used in the output files when a Subscriber is used. To change the delimiter character in the output files, edit the Output Transform style sheet. The delimiter character is assigned to a variable near the top of that style sheet.
Field Names	Specifies a comma-separated list of attribute names that can be referred to in the Schema Mapping rule. In the input files, the fields of the records must correspond to the order and positioning of the names in this list. See "Field Names" on page 29.
Object Class Name	Specifies the Novell® eDirectory™ class name that should be used when creating new objects to correspond to input files.
Allow Driver to Consume Its Own Output?	Prevents you from inadvertently creating a situation in which the driver writes output files that are immediately read in again as input of the same driver. The default is *No*. By default, the driver won't load if all the following conditions occur: ◆ You have both a Subscriber channel and a Publisher channel. ◆ The input and output directories are the same. ◆ The input and output file extensions are the same. If you want to feed the output of the Subscriber channel into the input of the Subscriber channel as a way to detect Identity Vault events to trigger other changes in the Identity Vault, set this parameter to *Yes*. For example, to update the Full Name attribute when the Given Name, Surname, or Initials attributes are updated, set this parameter to *Yes*.

Parameter	Description
Subscriber Options	
Output File Path	Specifies the directory on the local file system where output files will be created. An error occurs if this directory doesn't exist. See "Output File Path" on page 30.
Output File Extension	Output files have a unique name that ends with the characters in the *Output File Extension* parameter. If the output files from a Subscriber channel are used as input files for the Publisher channel of another Delimited Text driver, the destination file extension must match the source file extension parameter of the second driver.
Destination File Character Encoding (leave blank for default)	When this parameter contains no value, the default Java character encoding for your locale is used. To use an encoding other than the default for your locale, enter one of the canonical names from the Supported Encodings table (http://java.sun.com/j2se/1.4.2/docs/guide/intl/encoding.doc.html). The Publisher and Subscriber channels can use different character encodings.
Maximum Number of Transactions per Output File	Specifies the maximum number of transactions that are written to a single output file. When the file transaction limit is reached, the file closes, and a new file is created for subsequent transactions. To limit the number of transactions that can be written to a single file, leave this parameter blank or set it to zero. For more information, see the following item, *Maximum Time in Seconds Before Flushing All Transactions*.
Maximum Time in Seconds before Flushing All Transactions	If no new transactions have been written to the output file in the amount of time specified in this parameter, the file is closed. When new transactions need to be written, a new output file is created. If you don't want to limit the time that can pass before the output file is closed, leave this parameter blank or set it to zero.

Parameter	Description
Time of Day (Local Time) to Flush All Transactions	If a value is supplied for this parameter, the current output file is closed at the specified time each day. Subsequent transactions are written to a new file. This parameter does not prevent the *Maximum Number of Transactions per Output File* or the *Maximum Time in Seconds before Flushing All Transactions* parameters from also acting as output file thresholds. If you use this parameter and only want one file per day, set the other two parameters to zero. The format of this parameter can be HH:MM:SS (using the 24-hour clock) or H:MM:SS AM/PM. An hour is required, but the minutes and seconds are optional. Because the parameter assumes local time, any time zone information included in the value is ignored. The previous three parameters (*Maximum Number of Transactions per Output File*, *Maximum Time in Seconds before Flushing All Transactions*, and *Time of Day to Flush All Transactions*) are all capable of acting as a threshold for the transaction size a file is able to grow to, or for the time that it remains open to accept new transactions. As long as an output file is still open for writing by the Delimited Text driver, it shouldn't be considered as finalized. Avoid opening the file in any other process until the driver closes the file. For this reason, one of the three previous parameters must be set to assure that output files don't remain open indefinitely. To avoid this condition, if the driver detects that all three parameters are blank (or zero), it automatically sets the Maximum Number of Transactions per Output File to the value of 1.
Publisher Options	
Input File Path	The Publisher channel looks for new input files in the Input File Path, which is a directory on the local file system. Example paths: ◆ On Windows: `c:\csvsample\input` ◆ On Solaris and Linux: `/usr/lib/dirxml/rules/delim` ◆ For NetWare, you need to specify the volume (for example, `sys:csvsample\input`)
Input File Extension	The extension used to designate input files (for example, `csv`).
Source File Character Encoding (leave blank for default)	When this parameter contains no value, the default Java character encoding for your locale is used. To use an encoding other than the default for your locale, enter one of the canonical names from the Supported Encodings table (http://java.sun.com/j2se/1.4.2/docs/guide/intl/encoding.doc.html). Supported Encodings table (http://java.sun.com/j2se/1.4.2/docs/guide/intl/encoding.doc.html). If the Input File Extension parameter is `.xml`, the Source File Character Encoding can be indicated in one of two ways. For information, see "Source File Character Encoding" on page 33.

Parameter	Description
Rename File Extension	The Publisher channel uses only files that have the extension specified in the parameter. After the files have been processed, the value of the *Rename File Extension* parameter is appended to the filename, so the Publisher channel won't try to process the same file again. If the value of the Rename File Extension parameter is left blank, the source file is deleted after it is processed. **IMPORTANT**: If you change the default, use only characters that are valid in filenames on your platform. Invalid characters cause the rename to fail and the driver to reprocess the same file repeatedly.
Polling Rate (in Seconds)	When the Publisher channel has finished processing all source files, it waits the number of seconds specified in this parameter before checking for new source files to process.

B.1.7 ECMAScript

Enables you to add ECMAScript resource files. The resources extend the driver's functionality when Identity Manager starts the driver.

B.2 Global Configuration Values

Global configuration values (GCVs) enable you to specify settings for the Identity Manager features such as password synchronization and driver heartbeat, as well as settings that are specific to the function of an individual driver configuration. Some GCVs are provided with the drivers, but you can also add your own.

IMPORTANT: Password synchronization settings are GCVs, but it's best to edit them in the graphical interface provided on the Server Variables page for the driver, instead of the GCV page. The Server Variables page that shows Password Synchronization settings is accessible as a tab as with other driver parameters, or by clicking *Password Management* > *Password Synchronization*, searching for the driver, and clicking the driver name. The page contains online help for each Password Synchronization setting.

In iManager:

1. Click *Identity Manager* > *Identity Manager Overview*.
2. Click *Search* to search for the driver set that is associated with the driver.
3. Click the upper right corner of the driver icon.
4. Click *Edit Properties* > *Global Config Values*.

In Designer:

1. Open a project in the Modeler.
2. Right-click the driver icon or line, then select *Properties* > *Global Configuration Values.*

Table B-6 *Global Configuration Values > Password Configuration*

Option	Description
Application accepts passwords from Identity Manager	If True, allows passwords to flow from the Identity Manager data store to the connected system.
Identity Manager accepts passwords from application	If True allows passwords to flow from the connected system to Identity Manager.
Publish passwords to NDS password	Use the password from the connected system to set the non-reversible NDS® password in eDirectory.
Publish passwords to Distribution Password	Use the password from the connected system to set the NMAS™ Distribution Password used for Identity Manager password synchronization.
Require password policy validation before publishing passwords	If True, applies NMAS password policies during publish password operations. The password is not written to the data store if it does not comply.
Reset user's external system password to the Identity Manager password on failure	If True, on a publish Distribution Password failure, attempt to reset the password in the connected system by using the Distribution Password from the Identity Manager data store.
Notify the user of password synchronization failure via e-mail	If True, notify the user by e-mail of any password synchronization failures.
Connected System or Driver Name	The name of the connected system, application, or Identity Manager driver. This value is used by the e-mail notification templates.

B.3 Named Passwords

Identity Manager enables you to store multiple passwords securely for a particular driver. This functionality is referred to as Named Passwords. Each different password is accessed by a key, or name.

You can also use the Named Passwords feature to store other pieces of information securely, such as a user name. To configured Named Passwords, see Section 7.7, "Storing Driver Passwords Securely with Named Passwords," on page 48.

In iManager:

1. Click *Identity Manager > Identity Manager Overview*.
2. Click *Search* to search for the driver set that is associated with the driver.
3. Click the upper right corner of the driver icon.
4. Click *Edit Properties > Named Passwords*.

In Designer:

1. Open a project in the Modeler.
2. Right-click the driver line, then select *Properties > Named Passwords.*

B.4 Engine Control Values

The engine control values are a means through which certain default behaviors of the Metadirectory engine can be changed. The values can only be accessed if a server is associated with the Driver Set object.

In iManager:

1. Click *Identity Manager > Identity Manager Overview*.
2. Click *Search* to search for the driver set that is associated with the driver.
3. Click the upper right corner of the driver icon.
4. Click *Edit Properties > Engine Control Values*.

In Designer:

1. In the Modeler, right-click a driver line.
2. Select *Properties > Engine Control Values*.
3. Click the tooltip icon to the right of the *Engine Controls for Server* field. If a server is associated with the Identity Vault, the Engine Control Values display in the large pane.

Table B-7 *Engine Control Values*

Option	Description
Subscriber channel retry interval in seconds	The Subscriber channel retry interval controls how frequently the Metadirectory engine retries the processing of a cached transaction after the application shim's Subscriber object returns a retry status.
Qualified form for DN-syntax attribute values	The qualified specification for DN-syntax attribute values controls whether values for DN-syntax attribute values are presented in unqualified slash form or qualified slash form. A True setting means the values are presented in qualified form.
Qualified form from rename events	The qualified form for rename events controls whether the new-name portion of rename events coming from the Identity Vault are presented to the Subscriber channel with type qualifiers. For example, CN=. A True setting means the names are presented in qualified form.
Maximum eDirectory replication wait time in seconds	This setting controls the maximum time that the Metadirectory engine waits for a particular change to replicate between the local replica and a remote replica. This only affects operations where the Metadirectory engine is required to contact a remote eDirectory server in the same tree to perform an operation and might need to wait until some change has replicated to or from the remote server before the operation can be completed (for example, object moves when the Identity Manager server does not hold the master replica of the moved object; file system rights operations for Users created from a template.)

Option	Description
Use non-compliant backwards-compatible mode for XSLT	This control sets the XSLT processor used by the Metadirectory engine to a backwards-compatible mode. The backward-compatible mode causes the XSLT processor to use one or more behaviors that are not XPath 1.0 and XSLT 1.0 standards-compliant. This is done in the interest of backward compatibility with existing DirXML® style sheets that depend on the non-standard behaviors. For example, the behavior of the XPath "!=" operator when one operand is a node-set and the other operand is other than a node-set is incorrect in DirXML releases up to and including Identity Manager 2.0. This behavior has been corrected; however, the corrected behavior is disabled by default through this control in favor of backward compatibility with existing DirXML style sheets.
Maximum application objects to migrate at once	This control is used to limit the number of application objects that the Metadirectory engine requests from an application during a single query that is performed as part of a Migrate Objects from Application operation. If java.lang.OutOfMemoryError errors are encountered during a Migrate from Application operation, this number should be set lower than the default. The default is 50. **NOTE:** This control does not limit the number of application objects that can be migrated; it merely limits the batch size.
Set creatorsName on objects created in Identity Vault	This control is used by the Identity Manager engine to determine if the creatorsName attribute should be set to the DN of this driver on all objects created in the Identity Vault by this driver. Setting the creatorsName attribute allows for easily identifying objects created by this driver, but also carries a performance penalty. If not set, the creatorsName attribute defaults to the DN of the NCP™ Server object that is hosting the driver.
Write pending associations	This control determines whether the Identity Manager engine writes a pending association on an object during Subscriber channel processing. Writing a pending association confers little or no benefit but does incur a performance penalty. Nevertheless, the option exists to turn it on for backward compatibility.
Use password event values	This control determines the source of the value reported for the nspmDistributionPassword attribute for Subscriber channel Add and Modify events. Setting the control to False means that the current value of the nspmDistributionPassword is obtained and reported as the value of the attribute event. This means that only the current password value is available. This is the default behavior. Setting the control to True means that the value recorded with the eDirectory event is decrypted and is reported as the value of the attribute event. This means that both the old password value (if it exists) and the replacement password value at the time of the event are available. This is useful for synchronizing passwords to certain applications that require the old password to enable setting a new password.

Option	Description
Enable password synchronization status reporting	This control determines whether the Identity Manager engine reports the status of Subscriber channel password change events. Reporting the status of Subscriber channel password change events allows applications such as the Identity Manager User Application to monitor the synchronization progress of a password change that should be synchronized to the connected application.

B.5 Log Level

Each driver set and each driver has a log level field where you can define the level of errors that should be tracked. The level you indicate here determines which messages are available to the logs. By default, the log level is set to track error messages. (This also includes fatal messages.) To track additional message types, change the log level.

Novell® recommends that you use Novell Audit instead of setting the log levels. See the “*Identity Manager 3.5.1 Logging and Reporting*” guide.

In iManager:

1. Click *Identity Manager* > *Identity Manager Overview*.
2. Click *Search* to search for the driver set that is associated with the driver.
3. Click the upper right corner of the driver icon.
4. Click *Edit Properties* > *Log Level*.

In Designer:

1. Open a project in the Modeler.
2. Right-click the driver line, then select *Properties* > *Driver Log Level.*

Option	Description
Use log settings from the DriverSet	If this is selected, the driver logs events as the options are set on the Driver Set object.
Log errors	Logs just errors.
Log errors and warnings	Logs errors and warnings.
Log specific events	Logs the events that are selected. Click the icon to see a list of the events.
Only update the last log time	Updates the last log time.
Logging off	Turns logging off for the driver.
Turn off logging to DriverSet, Subscriber and Publisher logs	If selected, turns all logging off for this driver on the Driver Set object, Subscriber channel, and the Publisher channel.
Maximum number of entries in the log (50-500)	Number of entries in the log. The default value is 50.

B.6 Driver Image

Allows you to change the image associated with the driver. You can browse and select a different image from the default image.

The image associated with a driver is used by the Identity Manager Overview plug-in when showing the graphical representation of your Identity Manager configuration. Although storing an image is optional, it makes the overview display more intuitive.

NOTE: The driver image is maintained when a driver configuration is exported.

In iManager:

1 Click *Identity Manager* > *Identity Manager Overview*.
2 Click *Search* to search for the driver set that is associated with the driver.
3 Click the upper right corner of the driver icon.
4 Click *Edit Properties* > *Driver Image*.

In Designer:

1 Open a project in the Modeler.
2 Right-click the driver line, then select *Properties* > *iManager Icon.*

B.7 Security Equals

Use the Security page to view or change the list of objects that the driver is explicitly security equivalent to. This object effectively has all rights of the listed objects.

If you add or delete an object in the list, the system automatically adds or deletes this object in that object's "Security Equal to Me" property. You don't need to add the [Public] trustee or the parent containers of this object to the list, because this object is already implicitly security equivalent to them.

In iManager:

1 Click *Identity Manager* > *Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
2 Browse to the driver, then click the upper right corner of the driver icon.
3 Click *Edit Properties* > *Security Equals*.

Designer does not list the users the driver is security equals to.

B.8 Filter

Launches the Filter editor. You can edit the Filter from this tab.

In iManager:

1 Click *Identity Manager* > *Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
2 Browse to the driver, then click the upper right corner of the driver icon.

3. Click *Edit Properties > Filter*.

In Designer:

1. In an open project, click the *Outline* tab (Outline view).
2. Select the driver you want to manage the filter for, then click the plus sign to the left.
3. Double-click the *Filter* icon to launch the Filter Editor.

B.9 Edit Filter XML

Allows you to edit the filter directly in XML instead of using the Filter Editor.

In iManager:

1. Click *Identity Manager > Identity Manager Overview*, then click *Search* to search for the driver set that is associated with the driver.
2. Browse to the driver, then click the upper right corner of the driver icon.
3. Click *Edit Properties > Filter*.

In Designer:

1. In an open project, click the *Outline* tab (Outline view).
2. Select the driver you want to manage the filter for, then click the plus sign to the left.
3. Double-click the *Filter* icon to launch the Filter Editor, then click *XML Source* at the bottom of the Filter Editor.

B.10 Misc

Allows you to add a trace level to your driver. With the trace level set, DSTrace displays the Identity Manager events as the Metadirectory engine processes the events. The trace level affects only the driver it is set for. Use the trace level for troubleshooting issues with the driver when the driver is deployed. DSTRACE displays the output of the specified trace level.

In iManager:

1. Click *Identity Manager > Identity Manager Overview*.
2. Click *Search* to search for the driver set that is associated with the driver.
3. Click the upper right corner of the driver icon.
4. Click *Edit Properties > Misc*.

In Designer:

1. Open a project in the Modeler.
2. Right-click the driver line, then select *Properties > Trace.*

Option	Description
Trace level	Increases the amount of information displayed in DSTRACE. Trace level 1 shows errors, but not the cause of the errors. If you want to see password synchronization information, set the trace level to 5.

Option	Description
Trace file	When a value is set in this field, all Java information for the driver is written to the file. The value for this field is the path for that file. As long as the file is specified, Java information is written to this file. If you do not need to debug Java, leave this field blank.
Trace file size limit	Allows you to set a limit for the Java trace file. If you set the file size to Unlimited, the file grows in size until there is no disk space left.
Trace name	Driver trace messages are prepended with the value entered in this field.
Use setting from Driver Set	This option is only available in Designer. It allows the driver to use the same setting that is set on the Driver Set object.

B.11 Excluded Users

Use this page to create a list of users or resources that are not replicated to the application. Novell recommends that you add all objects that represent an administrative role to this list (for example, the Admin object).

In iManager:

1. Click *Identity Manager > Identity Manager Overview*.
2. Click *Search* to search for the driver set that is associated with the driver.
3. Click the upper right corner of the driver icon.
4. Click *Edit Properties > Excluded Users*.

Designer does not list the excluded users.

B.12 Driver Manifest

The driver manifest is like a resumé for the driver. It states what the driver supports, and includes a few configuration settings. The driver manifest is created by default when the Driver object is imported. A network administrator usually does not need to edit the driver manifest.

In iManager:

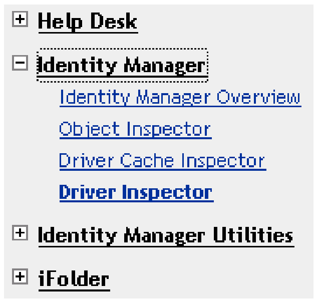
1. Click *Identity Manager > Identity Manager Overview*.
2. Click *Search* to search for the driver set that is associated with the driver.
3. Click the upper right corner of the driver icon.
4. Click *Edit Properties > Driver Manifest*.

In Designer:

1. Open a project in the Modeler.
2. Right-click the driver line, then select *Properties > Driver Manifest.*

B.13 Driver Cache Inspector

Figure B-2 *The Link to the Driver Cache Inspector*



The Driver Cache Inspector page uses a table format to display information about the cache file that stores events while the driver is stopped.

- **Driver:** A link to run the *Driver Overview* on the driver that is associated with this cache file.
- **Driver Set:** A link to run the *Driver Set Overview* on the driver set that holds the driver.
- **Driver's cache on:** Lists the server object that contains this instance of the cache file.
- **Start/Stop Driver icons:** Displays the current state of the driver and allows you to start or stop the driver.
- **Edit icon:** Enables you to edit the properties of the currently selected Server object.
- **Delete:** Deletes the selected items from the cache file.
- **Refresh:** Enables you to re-read the cache file and refresh the displayed information.
- **Show:** Limits the number of items to be displayed. The options are:
  - 25 per page
  - 50 per page
  - 100 per page
  - Other: Enables you to specify a desired number.
- **Actions:** Enables you to perform actions on the entries in the cache file. Click *Actions* to expand the menu, which includes:
  - **Expand All:** Expands all of the entries displayed in the cache file.
  - **Collapse All:** Collapses all of the entries displayed in the cache file.
  - **Go To:** Enables you to access a specified entry in the cache file. Specify the entry number, then click *OK*.
  - **Cache Summary:** Summarizes all events stored in the cache file.

B.14 Driver Inspector

The Driver Inspector page displays information about objects associated with the driver.

- **Driver:** A link to run the *Driver Overview* on the driver that is being inspected.
- **Driver Set:** A link to run the *Driver Set Overview* of the driver set that holds the driver.
- **Delete:** Deletes the associations of the selected objects.

- **Refresh:** Enables you to re-read all of the objects associated with the driver and refresh the displayed information.
- **Actions:** Enables you to perform actions on the objects associated with the driver. Click *Actions* to expand the menu, which includes:
    - **Show All Associations:** Displays all objects associated with the driver.
    - **Filter for Disabled Associations:** Displays all the driver's associated objects that have a Disabled state.
    - **Filter for Manual Associations:** Displays all the driver's associated objects that have a Manual state.
    - **Filter for Migrate Associations:** Displays all the driver's associated objects that have a Migrate state.
    - **Filter for Pending Associations:** Displays all the driver's associated objects that have a Pending state.
    - **Filter for Processed Associations:** Displays all the driver's associated objects that have a Processed state.
    - **Filter for Undefined Associations:** Displays all the driver's associated objects that have an Undefined state.
    - **Association Summary:** Displays the state of all objects associated with the driver.
- **Object DN:** Displays the DN of the associated objects.
- **State:** Displays the association state of the object.
- **Object ID:** Displays the value of the association.

B.15 Server Variables

This page lets you enable and disable Password Synchronization and the associated options for the selected driver.

When setting up Password Synchronization, consider both the settings on this page for an individual driver and the Universal Password Configuration options in your password policies.

This page lets you control which password Identity Manager updates directly, either the Universal Password for an Identity Vault, or the Distribution Password used for password synchronization by Identity Manager.

However, Novell Modular Authentication Service (NMAS) controls whether the various passwords inside the Identity Vault are synchronized with each other. Password Policies are enforced by NMAS, and they include settings for synchronizing Universal Password, NDS Password, Distribution Password, and Simple Password.

To change these settings in iManager:

1. In iManager, select *Passwords > Password Policies*.
2. Select a password policy, then click *Edit*.
3. Select *Universal Password*.

   This option is available from a drop-down list or a tab, depending on your version of iManager and your browser.
4. Select *Configuration Options*, make changes, then click *OK*.

NOTE: Enabling or disabling options on this page corresponds to values of True or False for certain global configuration values (GCVs) used for password synchronization in the driver parameters. Novell recommends that you edit them here in the graphical interface, instead of on the GCVs page. This interface helps ensure that you don't set conflicting values for the password synchronization GCVs.

Option	Description
Identity Manager accepts password (Publisher Channel)	If this option is enabled, Identity Manager allows passwords to flow from the connected system driver into the Identity Vault data store. Disabling this option means that no <`password`> elements are allowed to flow to Identity Manager. They are stripped out of the XML by a password synchronization policy on the Publisher channel. If this option is enabled, and the option below it for Distribution Password is disabled, a <`password`> value coming from the connected system is written directly to the Universal Password in the Identity Vault if it is enabled for the user. If the user's password policy does not enable Universal Password, the password is written to the NDS Password.
Use Distribution Password for password synchronization	To use this setting, you must have a version of eDirectory that supports Universal Password, regardless of whether you have enabled Universal Password in your password policies. If this option is enabled, a password value coming from the connected system is written to the Distribution Password. The Distribution Password is reversible, which means that it can be retrieved from the Identity Vault data store for password synchronization. It is used by Identity Manager for bidirectional password synchronization with connected systems. For Identity Manager to distribute passwords to connected systems, this option must be enabled. NMAS and Password policies control whether the Distribution Password is synchronized with other passwords in the Identity Vault. By default, the Distribution Password is the same as the Universal Password in the Identity Vault. If the password in the Identity Vault is to be independent of Password Synchronization, so that Identity Manager is a conduit only for synchronizing passwords among connected systems, change this default setting. In the Universal Password Configuration Options in a Password policy, disable *Synchronize Universal Password with Distribution Password*. This use of Identity Manager Password Synchronization is also referred to as “tunneling.”
Accept password only if it complies with user’s Password Policy	To use this setting, users must have a Password policy assigned that has Universal Password enabled, and Advanced Password Rules enabled and configured. If this option is chosen, Identity Manager does not write a password from this connected system to the Distribution Password in the Identity Manager data store or publish it to connected systems unless the password complies with the user's Password policy. By using the notification option that is also on this page, you can inform users when a password is not set because it is not compliant.

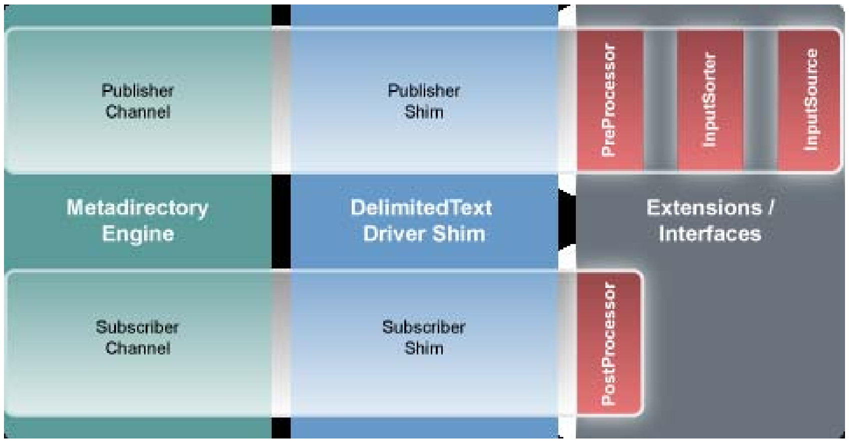
Option	Description
If password does not comply, ignore Password Policy on the connected system by resetting user's password to the Distribution Password	This option lets you enforce Password policies on the connected system by replacing a password that does not comply. If you select this option, and a user's password on the connected system does not comply with the user's Password policy, Identity Manager resets the password on the connected system by using the Distribution Password from the Identity Vault data store. Keep in mind that if you do not select this option, user passwords can become out-of-sync on connected systems. By using the notification option that is also on this page, you can inform users when a password is not set or reset. Notification is especially helpful for this option. If the user changes to a password that is allowed by the connected system but rejected by Identity Manager because of the Password policy, the user won't know that the password has been reset until the user receives a notification or tries to log in to the connected system with the old password. **NOTE:** Consider the connected system's password policies when deciding whether to use this option. Some connected systems might not allow the reset because they don't allow you to repeat passwords.
Always accept password; ignore Password Policies	If you select this option, Identity Manager does not enforce the user's Password policy for this connected system. Identity Manager writes the password from this connected system to the Distribution Password in the Identity Vault data store, and distributes it to other connected systems, even if the password does not comply with the user's Password policy.
Application accepts passwords (Subscriber Channel)	If you select this option, the driver sends passwords from the Identity Vault data store to this connected system. This also means that if a user changes the password on a different connected system that is publishing passwords to the Distribution Password in the Identity Vault data store, the password is changed on this connected system. By default, the Distribution Password is the same as the Universal Password in the Identity Vault, so changes to the Universal Password made in the Identity Vault are also sent to the connected system. If you want the password in the Identity Vault to be independent of Password Synchronization, so that Identity Manager is a conduit only for synchronizing passwords among connected systems, you can change this default setting. In the Universal Password Configuration Options in a password policy, disable *Synchronize Universal Password with Distribution Password*. This use of Password Synchronization is also referred to as "tunneling."
Notify the user of password synchronization failure via-email	If you select this option, e-mail is sent to the user if a password is not synchronized, set, or reset. The e-mail that is sent to the user is based on an e-mail template. This template is provided by the Password Synchronization application. However, for the template to work, you must customize it and specify an e-mail server to send the notification messages. **NOTE:** To set up e-mail notification, select *Passwords > Edit EMail Templates*.

Using the Delimited Text Driver Extensions

C

The Identity Manager Driver for Delimited Text defines different interfaces that you can implement in Java* classes to extend the base functionality of the driver.

Figure C-1 *Java Classes Interfaces*



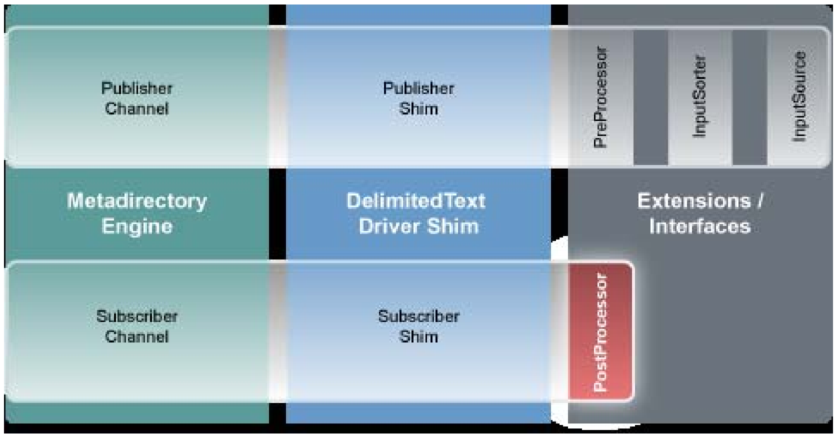
This section includes the following information about using the ImageFile and ImageSource extensions:

- Section C.1, "Using the ImageFile InputSource Extension," on page 111
- Section C.2, "Customizing ImageFile InputSource," on page 113
- Section C.3, "Using the ImageFile PostProcessor," on page 114
- Section C.4, "Customizing the ImageFile Extension," on page 116

C.1 Using the ImageFile InputSource Extension

The ImageFile InputSource extension allows you to easily import images of type GIF, JPG, and PNG into eDirectory™. The InputSource reads the image files from a specified directory, transforms them to a Base64-encoded string, and passes this string to the driver shim as if it were a normal delimited text string, together with additional information about the image, such as file size, file name, and modification date.

Figure C-2 InputSource Extension



Standard rules and style sheets, as used for other implementations of the Delimited Text driver, can be used to further process the image data. The image itself is meant to be stored in an attribute of syntax Octet String or Stream.

- Section C.1.1, "Installing the ImageFile InputSource Extension," on page 112
- Section C.1.2, "Configuring the Driver for the ImageFile InputSource Extension," on page 112

C.1.1 Installing the ImageFile InputSource Extension

The ImageFile InputSource extensions are installed when you install Identity Manager and select the Delimited Text driver. The extensions are configured in the driver parameters. Each extension takes a parameter string that is specified in the driver parameters and passes it to the extension during initialization. Extensions define their own format for the parameter string.

The extensions are included in the `DelimitedTextUtil.jar` file.

C.1.2 Configuring the Driver for the ImageFile InputSource Extension

Use the following table to customize your driver for the ImageFile extension.

Table C-1 Delimited Text Driver Parameters

Driver Parameter	XML Name	Sample Values	Purpose
Field Delimiter	field-delimiter	#	Indicates the character that is used to delineate field values in the input files. It must be one character. This must be set to the same character as the delim extension parameter or, if delim is not specified, to #.

Driver Parameter	XML Name	Sample Values	Purpose
Field Names (Field1, Field2, Field3…)	field-names	idx name prefix suffix size modified pic64	A comma-separated list of attribute names that can be referred to in the Schema Mapping rule. In the input files, the fields of the records must correspond to the order and positioning of the names in this list. For example, if you list eight field names in this parameter, then each record of the input files should have eight fields separated by the field delimiter character. The extension defines which fields it supports. The fields as listed here are the ones delivered by the extension. Altering the fields can cause malfunction.
InputSource Class	input-source	com.novell.nds.dirxml.driver.delimitedtext.imagefile	Class name of the input source.
InputSource init string	input-source-params	srcdir=c:\temp\dirxml;renameto=bak;delblacklist=false;renblacklistto=ignore;minsize=1000;debug=true	InputSource initialization parameters. For more information, see Section C.3.2, “Configuring the Driver for the ImageFile Extension,” on page 115.

C.2 Customizing ImageFile InputSource

You can fine-tune the behavior of the ImageFile InputSource by setting the parameters that are discussed in this section. Parameters are passed to the InputSource as a string.

The string must be in the following format:

```
<name1>=<value1>;<name2>=<value2>;<name n>=<value n>
```

Sample:

```
srcdir=c:\temp\dirxml;renameto=bak;delblacklist=false;renblacklist
to=ignore;minsize=1000;debug=true
```

Use the following values to configure the ImageFile properties:

Table C-2 *ImageFile InputSource Parameters*

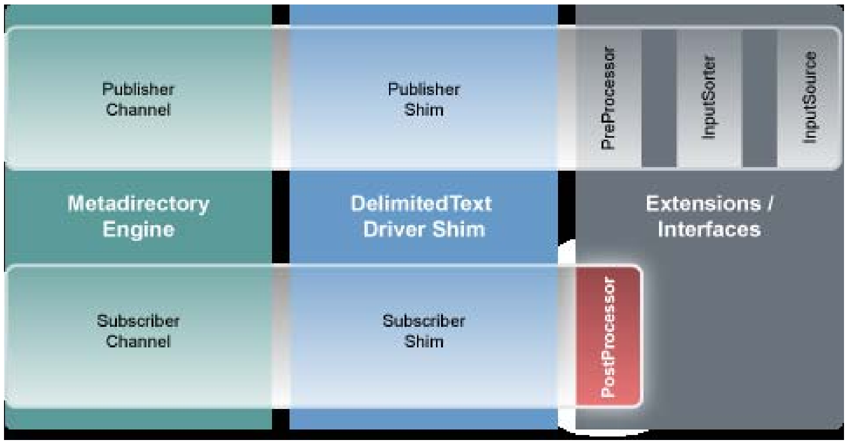
Parameter	Required	Default	Purpose
srcdir	yes		Directory where the image files are stored.
minsize	no	-1	Minimum size of the file to be processed. Number is the number of bytes. Set the value to -1 to disable the filter.

Parameter	Required	Default	Purpose
maxsize	no	-1	Maximum size of the file to be processed. Number is number of bytes. Set the value to -1 to disable the filter.
minwidth	no	-1	Minimum width of the image to be processed (number of pixels.) Set the value to -1 to disable the filter.
maxwidth	no	-1	Maximum width of the image to be processed (number of pixels.) Set the value to -1 to disable the filter.
minheight	no	-1	Minimum height of the image to be processed (number of pixels.) Set the value to -1 to disable the filter.
delim	no	#	Delimiter to be used in the created input files. The same delimiter must be configured in the driver parameters for the Delimited Text driver.
consolidate	no	true	Set consolidate to true to produce only one input file for all images. Set consolidate to false to produce an input file for each image.
renameto	no		Specify the suffix you want to be appended to the image files after processing. By default, the renameto parameter is not set and the files are deleted after processing.
debug	no	false	Set to true to turn on debug messages. Debugging is turned off by default.
delblacklist	no	true	Set to false to prevent deletion of the blacklist. The blacklist consists of all the image files that have been filtered out. By default, the blacklist is deleted.
renblacklistto	no		Specify any file suffix you want blacklist image files to be renamed to. By default the renameto parameter is not set and the processed files are handled according to the delblacklist parameter.

C.3 Using the ImageFile PostProcessor

The ImageFile PostProcessor extension allows you to easily export images from the directory to the file system as JPG or PNG files. The PostProcessor further processes the output file generated by the driver. It expects the output file to be in a certain format. The images are contained in the output file as Base64-encoded strings and are then converted into binary blobs and written to files in a specified directory.

Figure C-3 ImageFile PostProcessor Extension



Policies and style sheets can be used to control the export process. The images are usually stored in an attribute of syntax Octet String or Stream for binary data.

C.3.1 Installing the ImageFile PostProcessor Extension

The ImageFile PostProcessor extensions are installed when you install Identity Manager and select the Delimited Text driver. The extensions are configured in the driver parameters. Each extension takes a parameter string that is specified in the driver parameters and passes it to the extension during initialization. Extensions define their own format for the parameter string.

The extensions are included in the `DelimitedTextUtil.jar` file.

C.3.2 Configuring the Driver for the ImageFile Extension

- "Driver Module" on page 115
- "Driver Parameters" on page 115

Driver Module

The ImageFile InputSource requires the DelimitedText driver. Select Java and enter `com.novell.nds.dirxml.driver.delimitedtext.DelimitedTextDriver` as the driver module. If you want to run the driver with the Remote Loader, select *Connect to Remote Loader* and enter `com.novell.nds.dirxml.driver.delimitedtext.DelimitedTextDriver` as the the driver module in the Remote Loader console.

Driver Parameters

Use the following table to configure the driver. For additional information regarding other driver parameters, refer to the Identity Manager Driver for Delimited Text documentation. (http://www.novell.com/documentation/idm35drivers/delimited/data/bktitle.html)

Table C-3 ImageFile PostProcessor Parameters

Parameter	XML Name	Sample Value	Purpose
Field Delimiter	field-delimiter	#	Indicates the character that is used to delineate field values in the input files. It must be one character. This must be set to the same character as the delim extension parameter or, if delim is not specified, to #.
Field Names (Field1, Field2, Field3…)	field-names	idx name prefix suffix size modified pic64	A comma-separated list of attribute names that can be referred to in the Schema Mapping rule. In the input files, the fields of the records must correspond to the order and positioning of the names in this list. For example, if you list eight field names in this parameter, then each record of the input files should have eight fields separated by the field delimiter character. The extension defines which fields it supports. The fields as listed here are the ones delivered by the extension. Altering the fields can cause malfunction.
PostProcessor Class	post-processor	com.novell.nds.dirxml.driver.delimitedtext.imagefile	Class name of the input source.
PostProcessor init string	post-processor-params	destdir=/var/novell/idm/users/output/images;format=png;debug=true	PostProcessor initialization parameters. For more information, see Section C.1.1, "Installing the ImageFile InputSource Extension," on page 112.

C.4 Customizing the ImageFile Extension

You can fine-tune the behavior of the ImageFile extension by setting parameters. Parameters are passed to the PostProcessor as a string.

The string must be in the following format:

```
<name1>=<value1>;<name2>=<value2>;<name n>=<value n>
```

Sample:

```
destdir=/var/novell/idm/users/output/images;format=png;debug=true
```

Use the following table to customize the ImageFile extension.

Table C-4 *ImageFile Custom Parameters*

Parameter	Required	Default Value	Purpose
destdir	yes		Directory where the image files are stored.
delim	no	#	Used when parsing the output files. The same delimiter should be configured in the Delimited Text driver parameters.
format	no	png	Image format.
debug	no	false	Set the value to true to turn on debugging. Debugging is off by default.

Documentation Updates

D

The documentation was updated on the following dates:

- Section D.1, “March 5, 2008,” on page 119

D.1 March 5, 2008

Updates were made to the following sections. The changes are explained below.

D.1.1 Using the Delimited Text Driver Extensions

Location	Change
Section C.1.1, “Installing the ImageFile InputSource Extension,” on page 112	Changed the installation instructions.
Table C-2 on page 113	Changed the InputSource Class to com.novell.nds.dirxml.driver.delimitedtext.imagefile.
Section C.3.1, “Installing the ImageFile PostProcessor Extension,” on page 115	Changed the installation instructions.
Table C-3 on page 116	Changed the PostProcessor Class to com.novell.nds.dirxml.driver.delimitedtext.imagefile.