Identity Manager Driver for Entity Data Model

Implementation Guide

October 2019

Legal Notice

Contents

About this Guide and the Library

This guide explains how to install and configure the Identity Manager Driver for Entity Data Model.

Intended Audience

This book provides information for individuals responsible for implementing Identity Manager Driver for Entity Data Model.

Other Information in the Library

The library provides the following information resources:

Identity Intelligence Administrator Guide

Provides conceptual information and step-by-step guidance for administrative tasks in the Identity Intelligence product.

Identity Intelligence User Guide

Describes the user interface of the Identity Intelligence application and how you can use the features it offers.

1 Introduction

Entity data helps you to track user identities, accounts, and access rights in an enterprise or IT environment. For example, entity data provides the following information about a user:

- Organization based information, such as title and manager
- Accounts assigned to the user
- Access rights assigned to the user

Products such as Micro Focus Identity Intelligence leverage such entity data from Micro Focus Identity Manager to provide interactive and reporting capabilities. Identity Manager Driver for Entity Data Model collects various types of entity data such as identities, accounts, and access rights from Identity Manager and feeds this data to Identity Intelligence.

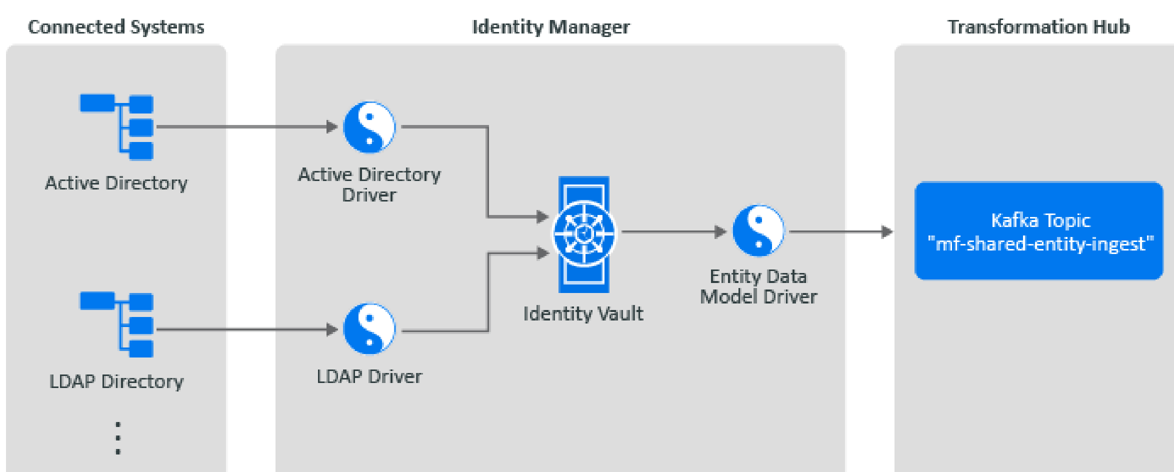
This chapter provides information about the following:

- "How the Driver Works" on page 7
- "Data Transfer Between Systems" on page 8

How the Driver Works

Identity Manager Driver for Entity Data Model enables you to track entity data like users, their account identifiers and access rights. Figure 1-1 on page 7 illustrates how the driver works to capture this information.

Figure 1-1 *Entity data flow*



- An entity object like a user account or a group is created, updated or deleted in an Identity Manager connected system like Active Directory.
- The driver for the connected system like Active Directory driver detects these *change events* on the entity objects and publishes to the Identity Vault.
- The Entity Data Model driver's policies on the driver's subscriber channel further receive and process these *entity change events* in *XML format*.

* The Entity Data Model Driver Shim receives the processed XML events, which converts the XML events into a *JSON format* using XSLT.
* The Entity Data Model Driver Shim then forwards the JSON formatted *entity change* data to the Kafka topic named *mf-shared-entity-ingest* in the Transformation Hub component of Identity Intelligence.
* The Entity Management Micro-service component of Identity Intelligence further processes the entity change data and persists the data in the entity tables in Vertica.
* The Identity Intelligence user interface reads the entity data that is persisted in Vertica using REST APIs, and presents to users for further analysis.

Data Transfer Between Systems

There are two data transfer channels between the Identity Vault and the connected application:

* **Publisher Channel:** Transfers data and events from the connected application to the Identity Vault. The Identity Manager Driver for Entity Data Model does not support this channel.
* **Subscriber Channel:** Transfers data and events from the Identity Vault to the connected application. Identity Manager Driver for Entity Data Model supports only data transfers from the Identity Vault to the Transformation Hub component of Identity Intelligence. Communication is one-way only.

  The Subscriber channel does the following:

  * Watches for additions and modifications to the Identity Vault objects.
  * Feeds these changes as JSON formatted messages to the Kafka topic named *mf-shared-entity-ingest* in the Transformation Hub component of Identity Intelligence, for further processing.

2 Implementation Checklist

Use the following checklist to verify that you complete the following tasks to have a complete solution with the driver.

	Checklist Items
❑	1. Review product overview information to learn about Identity Manager Driver for Entity Data Model. For more information, see Chapter 1, “Introduction,” on page 7.
❑	2. Ensure that you have installed the required software. For more information, see “System Requirements” on page 11.
❑	3. Ensure that you have completed the prerequisites steps. For more information, see “Prerequisites” on page 11.
❑	4. Determine where you want to install the driver shim. For more information, see “Planning the Driver Shim Installation” on page 12.
❑	5. Install the driver shim. For more information, see Chapter 4, “Installing the Driver Shim,” on page 15.
❑	6. Create and configure the driver. For more information, see, Chapter 5, “Creating and Configuring the Driver,” on page 17.
❑	7. Migrate the existing entity data from the Identity Vault to Identity Intelligence. For more information, see, Chapter 8, “Migrating Data from Identity Vault to Identity Intelligence,” on page 31.

3 Planning the Driver Installation

System Requirements

You need the following software to integrate Identity Manager with Identity Intelligence:

- Identity Manager 4.7.2
- Designer for Identity Manager 4.7.0
- Identity Intelligence 1.0

Prerequisites

Before installing the driver, ensure that you perform the following:

- Ensure that the drivers that are used with the Entity Data Model driver publish the values of the following attributes to the Identity Vault:
  - Attributes of the *User* class:
    - Given Name, Initials, Surname, Description, Internet EMail Address, L, Telephone Number, homePhone, mobile, photo, workforceID, Title, company, employeeStatus, employeeType and manager
  - Attributes of the *Group* class:
    - Description and Member
- Enable and configure ***account tracking*** in the Active Directory Driver:
  - The *DirXML-Accounts* attribute on an Identity Vault *User* object tracks information about accounts that a user has in different applications.
    - The Active Directory Driver maintains the values of the *DirXML-Accounts* attribute for the account identifiers that a user has in Active Directory.
  - The Entity Data Model driver uses the *DirXML-Accounts* attribute values to create and manage account records in Identity Intelligence.
  - To ensure that the *DirXML-Accounts* attribute is populated with appropriate values that can be used by the Entity Data Model driver, you must enable the account tracking GCV on the Active Directory Driver.
    - For detailed steps, see the Driver Properties > Global Configuration Values > Account Tracking section in the Identity Manager Driver for Active Directory Implementation Guide.

Planning the Driver Shim Installation

You can install the driver shim on either the Identity Manager system or a remote host. Figure 3-1 on page 12 illustrates the two installation options. The installation includes the following components:

- **Identity Vault:** Used by Identity Manager to store data for synchronization with Identity Intelligence. The Identity Vault is a persistent database powered by eDirectory. The vault can be viewed as a private data store for Identity Manager or as a metadirectory that holds enterprise-wide data. Data in the vault is available to any protocol supported by eDirectory, including NCP (the traditional protocol used by utilities, such as ConsoleOne and iManager), LDAP, and DSML.

  Since the Identity Vault is powered by eDirectory, you can easily integrate Identity Manager into your corporate directory infrastructure by using your existing directory tree as the vault. The Identity Vault runs on any platform supported by Identity Manager and communicates with the module on the connected system over a secure network link.
- **Entity Data Model Driver Shim:** Converts the XML based Identity Manager command and event language (XDS) to JSON messages required to integrate with Identity Intelligence. This driver uses a Java based driver shim (`EDMDriverShim.jar`). The driver shim is available on the Micro Focus Downloads website.
- **Remote Loader:** Enables a driver shim to execute outside of the metadirectory engine. The Remote Loader is typically used when a requirement of the driver shim is not met by the Identity Manager server. For example, if the metadirectory engine is running on Linux but you want to integrate with Active Directory, the remote loader is used to execute the Active Directory driver shim on a Windows server.

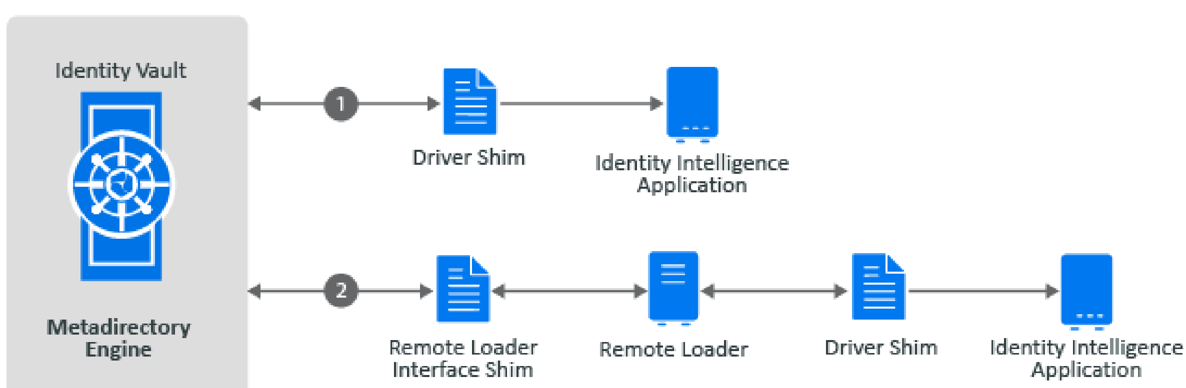
  The remote loader is a service that executes the driver shim and passes information between the shim and the metadirectory engine. You can install the driver shim on the server where the remote loader is running. You can choose to use SSL to encrypt the connection between the metadirectory engine and the Remote Loader.

  When you use the remote loader with the driver shim, two network connections are established:
  - Between Identity Manager and Remote Loader
  - Between Identity Intelligence and the driver shim

For more information on Remote Loader, see Deciding Whether to Use the Remote Loader in the NetIQ Identity Manager Driver Administration Guide.

The following figure illustrates the two options for installing the driver shim:

Figure 3-1 *Installing the driver shim*



Installing the Driver Shim on the Identity Manager System

The most common hosting for Identity Manager integration is in the Identity Vault metadirectory engine.

Advantages:

- The Integration module logs the trace messages in the metadirectory server trace log. Therefore, troubleshooting might be easier.
- No need to configure a remote loader instance.
- No extra TCP/IP traffic between the metadirectory and the remote loader.

Disadvantages:

- Resource consumption on the metadirectory server (memory, processor time).
- The requirement to restart the metadirectory server each time the integration module is installed or updated.

Installing the Driver Shim on a Remote System

The following are the advantages and disadvantages of the installing the driver shim on a remote system:

Advantages:

- Resource consumption (memory, processor time) is in a different process, or on another host.
- You need to restart only the remote loader process when the integration module is updated.

Disadvantages:

- Multiple trace files. Therefore, when troubleshooting, you might need to examine trace files from both the metadirectory process and the remote loader process.
- The need to configure a remote loader instance.
- Extra TCP/IP traffic between the metadirectory and the remote loader.

4 Installing the Driver Shim

Before you create and configure the driver, you need to install the driver shim in order to be able to create and configure the driver.

- "Installing on Linux as a Root User and Windows" on page 15
- "Installing on Linux as a Non-Root User" on page 15

Installing on Linux as a Root User and Windows

1 Download the `NIdM_Integration_Module_<version>_EntityDataModel.zip` file from the Micro Focus Downloads website and extract the contents of the ZIP file to a temporary location on your server, which is either the Identity Manger server or the Remote Loader, depending on where you want to install the driver shim.

2 (Conditional) If you want to install the driver locally on the Identity Manager server, stop the Identity Vault.

3 (Conditional) If you want to install the driver on the Remote Loader, stop the Remote Loader instance.

4 Perform the following:

- **Linux**: Log in to your server as `root` and run the following command in a command prompt:

  ```
  rpm -Uvh <Extracted ZIP File Temporary Location>/linux/netiq-DXMLedm.rpm
  ```

  This will place the files automatically in the `/opt/novell/eDirectory/lib/dirxml/classes` location with all the required permissions.
- **Windows**: Navigate to the `<Extracted ZIP File Temporary Location>\windows` folder and copy the following files to `<Identity Manager installation>\eDirectory\lib` or `<Identity Manager installation>\RemoteLoader\lib` folder:
  - `EDMDriverShim.jar`
  - `kafka-clients-<version>.jar`

5 (Conditional) If you installed the driver locally on the Identity Manager server, start the Identity Vault.

6 (Conditional) If you installed the driver on the Remote Loader, start the Remote Loader instance.

Installing on Linux as a Non-Root User

This section provides information on how to install the driver files on Linux.

To install the driver files as a non-root user:

1 Verify that `<non-root eDirectory location>/rpm` directory exists and contains `_db.000` file.

The `_db.000` file is created during a non-`root` installation of the Identity Manager engine. Absence of this file might indicate that Identity Manager is not properly installed. Reinstall Identity Manager to correctly place the file in the directory.

2 To set the `root` directory to the location of non-`root` Identity Vault, enter the following command in the command prompt:

```
ROOTDIR=<non-root eDirectory location>
```

This will set the environmental variables to the directory where Identity Vault is installed as a non-`root` user.

3 To install the driver files, enter the following command:

```
rpm --dbpath $ROOTDIR/rpm -Uvh --relocate=/usr=$ROOTDIR/opt/novell/eDirectory
--relocate=/etc=$ROOTDIR/etc --relocate=/opt/novell/eDirectory=$ROOTDIR/opt/
novell/eDirectory --relocate=/opt/novell/dirxml=$ROOTDIR/opt/novell/dirxml --
relocate=/var=$ROOTDIR/var --badreloc --nodeps --replacefiles <rpm-location>
```

For example, to install the Entity Data Model driver RPM, use this command:

```
rpm --dbpath $ROOTDIR/rpm -Uvh --relocate=/usr=$ROOTDIR/opt/novell/eDirectory
--relocate=/etc=$ROOTDIR/etc --relocate=/opt/novell/eDirectory=$ROOTDIR/opt/
novell/eDirectory --relocate=/opt/novell/dirxml=$ROOTDIR/opt/novell/dirxml --
relocate=/var=$ROOTDIR/var --badreloc --nodeps --replacefiles /home/user/
netiq-DXMLedm.rpm
```

where `/opt/novell/eDirectory` is the location where non-`root` eDirectory is installed and `/home/user/` is the home directory of the non-`root` user.

After the driver shim files are installed, create the driver. For more information, see Chapter 5, “Creating and Configuring the Driver,” on page 17.

5 Creating and Configuring the Driver

After you install the driver shim on the server where you want to run the driver, you must create the driver in the Identity Vault by using Designer.

Importing the Driver Packages

The driver packages contain the items required to create a driver, such as policies, filters, and Schema Mapping policies. Before you create the driver, import the latest packages to Designer. This driver requires the following packages:

- `NETQEDMBASE_<version>.jar`
- `NETQEDMCFG_<version>.jar`

To import packages into the package catalog:

1. Download the `NIdM_Integration_Module_<version>_EntityDataModel.zip` file from the Micro Focus Downloads website and extract the contents of the ZIP file to a temporary location on your Designer machine.
2. Select the package catalog object in the Outline view, then right-click and select **Import Package**.
3. Click **Browse**, then browse to and select the package JAR files from the `<Extracted ZIP File Temporary Location>\packages` directory on the file system.
4. Click **OK** to import the package.

Creating the Driver Object

This section helps you configure the Entity Data Model driver and establish its basic settings.

1. Open Designer.

   NOTE: Ensure that the *Common Settings* package is installed in the **Package Catalog** before you create the driver object.

   ---

2. Right-click the driver, select **New** > **Driver**. The Driver Configuration Wizard opens.
3. Select **Entity Data Model Base**, then click **Next**.
4. In the **Select Mandatory Features** page, select **Default Configuration** and click **Next**.
5. If you are using Designer version 4.7.0 or later, click **Next**. If not, you must upgrade Designer to version 4.7.0 or later, and start again from step 1.

6 For **Driver Name**, specify a value and click **Next** to proceed. The default driver name is `Entity Data Model`.

7 In the **Driver Options**, specify the following details, and then click **Next**:

- **Use SSL**: Select an appropriate value to indicate if a secure SSL connection is required between the driver and the Transformation Hub Kafka cluster.

  If you select **Yes**, create a KeyStore file that contains the Transformation Hub's CA certificate in a temporary directory on the computer where the driver is being installed. For detailed steps, see Creating a KeyStore.

  After creating the KeyStore file, specify the following additional details:

  - **KeyStore Path for SSL certs**: Specify the full path to the KeyStore file that contains the Transformation Hub's CA certificate.
  - **KeyStore Password**: Specify the password used to access the KeyStore file that contains the Transformation Hub's CA certificate.

- **Kafka Server Hosts and Port Numbers**: Specify a comma-separated list of hostnames (fully qualified domain names) and ports for establishing communication with the Transformation Hub Kafka cluster. The default SSL port is *9093* and the default non-SSL port is *9092*.

  Not all servers in the cluster must be listed, but if none of the servers listed can be contacted, the driver cannot send data to the Transformation Hub. Specify at least one server.

  For example, `kafka1.example.com:9092` or `kafka1.example.com:9092,kafka2.example.com:9092`

  NOTE: Ensure that the FQDNs of the Transformation Hub Kafka nodes resolve successfully from the Identity Manager Server or Remote Loader where the driver is installed.

- **Kafka Topic Name**: Specify the name of the Kafka topic to which the entity data will be sent as *mf-shared-entity-ingest*.

  NOTE: It is recommended to change the topic name only if you want to send the entity data to your own Kafka cluster outside of Identity Intelligence.

- (Conditional) **Advanced Kafka Properties**: Specify the advanced properties for the Kafka connection.

  IMPORTANT: Specify these properties at your own discretion and validate them because the changes are applied as is. For more information about these properties, see the *Producer Configs* section in the Kafka documentation.

8 (Conditional) Fill in the following fields for Remote Loader information:

- **Connect To Remote Loader:** Select **Yes** or **No** to determine if the driver will use the Remote Loader. If you select **No**, skip to Step 9. If you select **Yes**, use the following information to complete the Remote Loader configuration.
- **Host Name:** Specify the host name or IP address of the server where the driver’s Remote Loader service is running.
- **Port:** Specify the port number where the Entity Data Model driver instance is configured in the Remote Loader. The default port number is `8090`.

- **KMO:** Specify the key name of the Key Material Object (KMO) that contains the keys and certificate the Remote Loader uses for an SSL connection. This parameter is only used when you use SSL and mutual authentication for connections between the Remote Loader and the Identity Manager engine.
- **Other Parameters:** Specify any other parameters required to connect to the Remote Loader. Any parameters specified must use a key-value pair format, as follows: `paraName1=paraValue1 paraName2=paraValue2`.
- **Remote Password:** Specify the Remote Loader's password as defined on the Remote Loader. The Identity Manager server (or Remote Loader) requires this password to authenticate to the Remote Loader.
- **Driver Password:** Specify the driver object password that is defined in the Remote Loader service. The Remote Loader requires this password to authenticate to the Identity Manager server.

9 Click **Next**.

10 Review the summary of tasks that will be completed to create the driver, then click **Finish**.

11 (Conditional) If you want Identity Intelligence to do user reconciliation on any user attribute that is not present by default in the Entity Data Model identity schema and the Driver Filter:

- You must add the attribute in the **Driver Filter** under the **User** class. For detailed steps, see Controlling the Flow of Objects with the Filter.
- In the driver's Schema Map Policy, you must add an attribute row under the **User** class. In the attribute row, specify the **Identity Vault** attribute as the user attribute and the Application attribute as `entity_reconciliation_id`. For detailed steps, see Defining Schema Map Policies.

For example, if you want Identity Intelligence to do user reconciliation on the `Full Name` attribute, you must update the Driver Filter and the Schema Map Policy as indicated in the following XML source snippets:

- **Driver Filter:**

```
<filter-attr attr-name="Full Name" merge-authority="edir"
publisher="ignore" publisher-optimize-modify="true" subscriber="sync"/>
```

- **Schema Map Policy:**

```
<attr-name class-name="User">
      <nds-name>Full Name</nds-name>
      <app-name>entity_reconciliation_id</app-name>
</attr-name>
```

Deploying the Driver Object

After you create, configure, or modify the driver, you must deploy the driver to the Identity Vault, because Designer is an offline tool.

1 In Designer, open your project.

2 To deploy only the target driver, in the Modeler view right-click the driver line, then select **Live > Deploy**.

3 If you are authenticated to the Identity Vault, skip to Step 5; otherwise, specify the following information to authenticate:

- **Host**: Specify the IP address or DNS name of the server hosting the Identity Vault.

- **Username**: Specify the DN of the user object used to authenticate to the Identity Vault.
- **Password**: Specify the user's password.

4 Click **OK**.

5 Read through the deployment summary, then click **Deploy**.

6 Click **OK**.

7 Click **Define Security Equivalence** to assign rights to the driver.

The driver requires rights to objects within the Identity Vault. The *Admin* user object is most often used to supply these rights. However, you might want to create a *DriversUser* (for example) and assign security equivalence to that user.

7a Click **Add**, then browse to and select the object with the correct rights.

7b Click **OK** twice.

For more information about defining a Security Equivalent User in objects for drivers in the Identity Vault, see Establishing a Security Equivalent User in the NetIQ Identity Manager Security Guide.

8 Click **Exclude Administrative Roles** to exclude users that should not be synchronized.

You should exclude any administrative User objects, such as *Admin* and *DriversUser* from synchronization.

8a Click **Add**, then browse to and select the user object you want to exclude.

8b Click **OK**.

8c Repeat Step 8a and Step 8b for each object you want to exclude.

8d Click **OK**.

9 Click **OK**.

Starting the Driver

When a driver is created, it is stopped by default. To make the driver work, you must start the driver. Identity Manager is an event-driven system, so after the driver is started, it will not do anything until an event occurs.

To start the driver:

1 If you are using the Remote Loader with the driver, make sure the Remote Loader driver instance is running.

2 In Designer, open your project.

3 In the Modeler view, right-click the driver icon or the driver line, then select **Live > Start Driver**.

For instructions about starting and stopping the driver instance in the Remote Loader on Linux and Windows, see Starting and Stopping the Remote Loader in the NetIQ Identity Manager Driver Administration Guide.

Verifying the Functionality

After you deploy and configure the driver, you need to verify that the driver correctly creates and updates entity data.

1. Ensure that you have started the driver.
2. Create a test user in the Identity Vault.
3. Verify that a corresponding user is found in **Users & Entities** > **Search**.

   For more information, see "Exploring User Profiles" in the Identity Intelligence User Guide.
4. If your Identity Vault already contains User objects, you can use **Migrate from Identity Vault** in iManager to validate the configuration.

6 Upgrade Procedure

The driver upgrade process involves upgrading the installed driver packages and updating the driver files.

This section provides general instructions for updating a driver. For information about updating the driver to a specific version, search for that driver patch in the Micro Focus Patch Finder Download Page and follow the instructions from the *Readme* file accompanying the driver patch release.

Upgrading the Installed Packages

1 Download the latest available packages.

   To configure Designer to automatically read the package updates when a new version of a package is available, click **Windows** > **Preferences** > **NetIQ** > **Package Manager** > **Online Updates** in Designer. However, if you need to add a custom package to the Package Catalog, you can import the package `.jar` file. For more information about creating custom packages, see Upgrading Installed Packages in NetIQ Designer for Identity Manager Administration Guide.

2 Upgrade the installed packages.

   2a Open the project containing the driver.

   2b Right-click the driver for which you want to upgrade an installed package, then click **Driver** > **Properties**.

   2c Click **Packages**.

   If there is a newer version of a package, there is check mark displayed in the Upgrades column.

   2d Click **Select Operation** for the package that indicates there is an upgrade available.

   2e From the drop-down list, click **Upgrade**.

   2f Select the version that you want to upgrade to, then click **OK**.

   NOTE: Designer lists all versions available for upgrade.

   2g Click **Apply**.

   2h (Conditional) Fill in the fields with appropriate information to upgrade the package, then click **Next**.

   Depending on which package you selected to upgrade, you must fill in the required information to upgrade the package.

   2i Read the summary of the packages that will be installed, then click **Finish**.

   2j Review the upgraded package, then click **OK** to close the Package Management page.

   For detailed information, see the Upgrading Installed Packages in NetIQ Designer for Identity Manager Administration Guide.

Applying the Driver Patch

The driver patch updates the driver files. You can install the patch as a `root` or `non-root` user.

Prerequisites

Before installing the patch, complete the following steps:

1. Take a back-up of the current driver configuration.
2. (Conditional) If the driver is running with the Identity Manager engine, stop the Identity Vault and the driver instance.
3. (Conditional) If the driver is running with a Remote Loader instance, stop the Remote Loader instance and the driver instance.
4. In a browser, navigate to the Micro Focus Patch Finder Download Page and search for the driver patch.
5. Download and unzip the contents of the patch file to a temporary location on your server.

Applying the Patch on Linux as a Root User and Windows

In a root installation, the driver patch installs the driver files in the default locations on Linux. On Windows, you need to manually copy the files to the default locations.

1. Update the driver files:
   - **Linux**: Log in to your server as root and run the following command in a command prompt:

     `rpm -Uvh <`*`Extracted Driver Patch File Temporary Location`*`>/linux/netiq-DXMLedm.rpm`
   - **Windows**: Navigate to the `<`*`Extracted Driver Patch File Temporary Location`*`>\windows` folder and copy the following files to `<`*`Identity Manager installation`*`>\eDirectory\lib or <`*`Identity Manager installation`*`>\RemoteLoader\lib` folder.
     - `EDMDriverShim.jar`
     - `kafka-clients-<`*`version`*`>.jar`
2. (Conditional) If the driver is running locally, start the Identity Vault and the driver instance.
3. (Conditional) If the driver is running with a Remote Loader instance, start the Remote Loader instance and the driver instance.

Applying the Patch as a Non-Root User

1. Verify that `<`*`non-root eDirectory location`*`>/rpm` directory exists and contains the file, `_db.000`.

   The `_db.000` file is created during a non-`root` installation of the Identity Manager engine. Absence of this file might indicate that Identity Manager is not properly installed. Reinstall Identity Manager to correctly place the file in the directory.
2. To set the `root` directory to non-`root` eDirectory location, enter the following command in the command prompt:

   `ROOTDIR=<`*`non-root eDirectory location`*`>`

   This will set the environmental variables to the directory where eDirectory is installed as a non-`root` user.
3. Download the patch and untar or unzip the downloaded file.
4. To install the driver files, enter the following command:

```
rpm --dbpath $ROOTDIR/rpm -Uvh --relocate=/usr=$ROOTDIR/opt/novell/eDirectory-
-relocate=/etc=$ROOTDIR/etc --relocate=/opt/novell/eDirectory=$ROOTDIR/opt/
novell/eDirectory --relocate=/opt/novell/dirxml=$ROOTDIR/opt/novell/dirxml --
relocate=/var=$ROOTDIR/var --badreloc --nodeps --replacefiles <rpm-location>
```

For example, to install the Entity Data Model driver RPM, use this command:

```
rpm --dbpath $ROOTDIR/rpm -Uvh --relocate=/usr=$ROOTDIR/opt/novell/eDirectory
--relocate=/etc=$ROOTDIR/etc --relocate=/opt/novell/eDirectory=$ROOTDIR/opt/
novell/eDirectory --relocate=/opt/novell/dirxml=$ROOTDIR/opt/novell/dirxml --
relocate=/var=$ROOTDIR/var --badreloc --nodeps --replacefiles /home/user/
netiq-DXMLedm.rpm
```

where `/opt/novell/eDirectory` is the location where non-`root` eDirectory is installed and `/home/user/` is the home directory of the non-`root` user.

7 Understanding the Schema Mapping

The Schema Mapping Policy of the Entity Data Model Driver maps class names and attribute names between the Identity Vault namespace and the Entity Data Model namespace.

The following table describes the attributes mapping in detail for *User* class:

Identity Vault	Entity Data Model	Comments
Class: User	Class: identity	
GUID	entity_producer_id	
Given Name	identity_name_given	
Initials	identity_name_middle	
Surname	identity_name_family	
Description	identity_notes	
Internet EMail Address	identity_email	
L	identity_location	
Telephone Number	identity_phone_office	
homePhone	identity_phone_home	
mobile	identity_phone_mobile	
photo	identity_photo	
workforceID	persona_id	
Title	persona_title	
company	persona_organization	
employeeStatus	persona_status	
employeeType	persona_type	
manager	identity_manager	*Manager-direct report* relation between two User objects is derived from this attribute.
DirXML-Accounts		Relation between User object and its associated accounts is derived from this attribute
nrfAssignedRoles		Relation between User object and its assigned Roles is derived from this attribute.
nrfAssignedResources		Relation between User object and its assigned Resources is derived from this attribute.

The following table describes the attributes mapping in detail for *Group* class:

Identity Vault	Entity Data Model	Comments
Class: Group	Class: identitygroup	
Attributes/Metadata:		
GUID	entity_producer_id	
qualified-src-dn	identitygroup_id, identitygroup_name	Both *identitygroup_id* and *identitygroup_name* are derived from *qualified-src-dn*.
Description	identitygroup_description	
Member	identity_member	*Member of* relation between User and Group objects is derived from this attribute.
nrfAssociatedRoles		Relation between Group object and its assigned Roles is derived from this attribute.

The following table describes the attributes mapping in detail for *nrfRole* class:

Identity Vault	Entity Data Model	Comments
Class: nrfRole	Class: entitlement	
Attributes/Metadata:		
GUID	entity_producer_id	
nrfLocalizedNames	entitlement_name	
nrfLocalizedDescrs	entitlement_description	
qualified-src-dn	entitlement_id	*entitlement_id* is derived from *qualified-src-dn*.
nrfChildRoles		*Parent-child* relation between two Role objects is derived from this attribute.

The following table describes the attributes mapping in detail for *nrfResource* class:

Identity Vault	Entity Data Model	Comments
Class: nrfResource	Class: entitlement	
Attributes/Metadata:		
GUID	entity_producer_id	
nrfLocalizedNames	entitlement_name	
nrfLocalizedDescrs	entitlement_description	
qualified-src-dn	entitlement_id	*entitlement_id* is derived from *qualified-src-dn*.
nrfEntitlementRef		Relation between Resource and Entitlement objects is derived from this attribute.

The following table describes the attributes mapping in detail for *DirXML-Entitlement* class:

Identity Vault	Entity Data Model	Comments
Class: DirXML-Entitlement	Class: entitlement	
Attributes/Metadata:		
GUID	entity_producer_id	
XmlData	entitlement_name	The *entitlement_name* attribute is derived from the *display-name* attribute of *XmlData*.
XmlData	entitlement_description	The *entitlement_ description* attribute is derived from the *description* attribute of *XmlData*.
qualified-src-dn	entitlement_id	*entitlement_id* is derived from *qualified-src-dn*.

The following table describes the attributes mapping in detail for *nrfResourceAssociation* class:

Identity Vault	Entity Data Model	Comments
Class: nrfResourceAssociation	Class: entitlement	
Attributes:		
nrfRole	-	For the relation between Role and Resource objects, the GUID of the Role object is derived from the *nrfRole* attribute.
nrfResource	-	For the relation between Role and Resource objects, the GUID of the Resource object is derived from the *nrfResource* attribute.

8 Migrating Data from Identity Vault to Identity Intelligence

If you are deploying the driver to an Identity Vault with existing entity data that is, Users, Groups, Roles, Resources, Entitlements and relations between them, you need to perform an initial migration of the Identity Vault data into Identity Intelligence.

You can migrate the Identity Vault's entity data with or without relations. The possible relations in the entity data among Users, Groups, Roles, Resources and Entitlements are as follows:

- *Manager-direct report* relation between two User objects
- Relation between User object and its associated accounts
- Relation between User object and its assigned Roles
- Relation between User object and its assigned Resources
- *Member of* relation between User and Group objects
- Relation between Group object and its assigned Roles
- *Parent-child* relation between two Role objects
- Relation between Resource and Entitlement objects
- Relation between Role and Resource objects

Migrating Data without Relations

If your Identity Vault entity data does not have relations between the entity objects, the migration is a one-step process.

You can either manually select the User, Group, Role, Resource and Entitlement objects to be migrated by using the **Migrate from Identity Vault** option in iManager, or allow the Identity Vault to automatically submit all objects by using the **Synchronize** option in iManager.

Migrating Data with Relations

If your Identity Vault entity data has relations between the entity objects, the migration is a two-step process.

1. Perform the steps mentioned in "Migrating Data without Relations" on page 31.
2. After the migration is complete, repeat the procedure. Repeating the procedure ensures that relations that could not be established in the first step are resolved.

   For example, if an employee's Identity Vault object is synchronized before the employee's manager's Identity Vault object, the manager relation cannot be established in Identity Intelligence because the manager's object does not exist in Identity Intelligence yet. When you repeat the process, migration occurs after all objects are created in Identity Intelligence, so that all relations can be established in Identity Intelligence.

NOTE: Data migration may take a while depending on the number of entities. If there are a large number of entities, ensure that your system meets the requirements specified for Large Workload.

A Creating a KeyStore

To enable a secure SSL connection between the driver and the Transformation Hub Kafka cluster, you must create a KeyStore file that contains the Transformation Hub's CA certificate as follows:

1. Obtain the Transformation Hub's CA certificate as described in the “Obtaining the Transformation Hub Certificate” section of the Administrator Guide for Identity Intelligence.
2. Copy the certificate to a temporary directory on the computer where the driver is being installed.
3. On the computer where the driver is being installed, import the certificate into a KeyStore file that the driver can use:

   3a Use the `keytool` utility which is found in the `jre/bin` directory.

   For example, if you want to import the certificate saved as `kafka.cer` into a new KeyStore file named `keystore.jks` in the current directory, enter the following in the command line:

   ```
   keytool -import -file kafka.cer -alias <alias> -keystore keystore.jks
   ```

   3b Specify the KeyStore password.

   3c When you are asked to trust the certificate, specify **Yes**, then click **Enter**.

B Known Issues

Micro Focus strives to ensure our products provide quality solutions for your enterprise software needs. The following issues are currently being researched. If you need further assistance with any issue, please contact Technical Support.

Identity Information Moves to the Rejected Table when the Identity's Photo Size is Too Large

Issue: If the photo for an ingested identity is larger than 65 KB, Identity Intelligence sends the respective identity information to the rejected table and does not display the user. `(Bug 1140568)`

Workaround: Ensure that the identity's photo size is less than 65 KB. If the photo size is more than 65 KB, you can compress the photo and then update.

Exception Reported when Running Entity Data Model Driver and Google Apps Driver on the Same Server

Issue: If the Entity Data Model Driver and Google Apps Driver are running on the same server and **Override JAXP Factory** is set to *true* on the Google Apps Driver, the Entity Data Model Driver reports the following error while processing events on the Subscriber channel:

```
DirXML Log Event -------------------
     Driver:   \IDM47_TREE\system\driverset1\Entity Data Model
     Channel:  Subscriber
     Object:   \IDM47_TREE\data\employees\Administrative\Albert Monday
     Status:   Error
     Message:  Code(-9010) An exception occurred:
javax.xml.transform.TransformerFactoryConfigurationError: Provider
javax.xml.transform.sax.SAXTransformerFactory could not be instantiated:
java.lang.IllegalAccessException: Class javax.xml.transform.FactoryFinder can not
access a member of class javax.xml.transform.sax.SAXTransformerFactory with
modifiers "protected"
```

`(Bug 1145268)`

Workaround: Set **Override JAXP Factory** to *false* on the Google Apps Driver, or run the Entity Data Model Driver and Google Apps Driver on different servers.