AUTHORIZED DOCUMENTATION

Identity Manager 3.6.1 Staging Best Practices Guide

Novell®

Identity Manager™

3.6.1

June 24, 2010

www.novell.com

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 www.novell.com/documentation.

Contents

About This Guide

Welcome to Novell Identity Manager 3.6.1 Staging Best Practices Guide. This guide provides step-by-step procedures to move your Identity Management solutions from one stage to the subsequent stages.

This guide introduces the following:

- Chapter 1, "Overview," on page 9
- Chapter 2, "Preparing for Staging," on page 11
- Chapter 3, "Staging a Project," on page 17
- Chapter 4, "Best Practices in Moving Objects Across Stages," on page 25

Audience

The guide is intended for Identity Manager consultants and customers.

Feedback

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

Documentation Updates

For the most recent version of the *Novell Identity Manager 3.6.1 Staging Best Practices Guide*, visit the Identity Manager Web site (http://www.novell.com/identity/access/identity_integration).

Additional Documentation

- Understanding Designer for Identity Manager (http://www.novell.com/documentation/designer30/designer_intro/data/front.html)
- Identity Manager 3.6.1 Installation Guide (http://www.novell.com/documentation/idm36/idm_install/data/front.html)
- Understanding Policies for Identity Manager 3.6 (http://www.novell.com/documentation/idm36/policy/data/bookinfo.html)
- Policies in Designer 3.5 (http://www.novell.com/documentation/idm36/policy_designer/data/bookinfo.html)
- Novell Credential Provisioning for Identity Manager 3.6 (http://www.novell.com/documentation/idm36/policy_credprov/data/bookinfo.html)
- Identity Manager 3.6 DTD Reference (http://www.novell.com/documentation/idm36/policy_dtd/data/bookinfo.html)
- Identity Manager 3.6 Driver Guides (http://www.novell.com/documentation/idm36drivers/index.html)

Overview

1

1.1 What is Staging

Software products need testing before they are deployed in an IT environment. Staging is a process of development and testing software products. Staging provides users the flexibilty to validate applications in real time to ensure uniformity across all stages.

Identity Manager staging is similar to any software deployment process. Nearly all the Identity Manager projects operate in more than two stages, such as development, testing, and the production environment.

Figure 1-1 *Staging Movement*

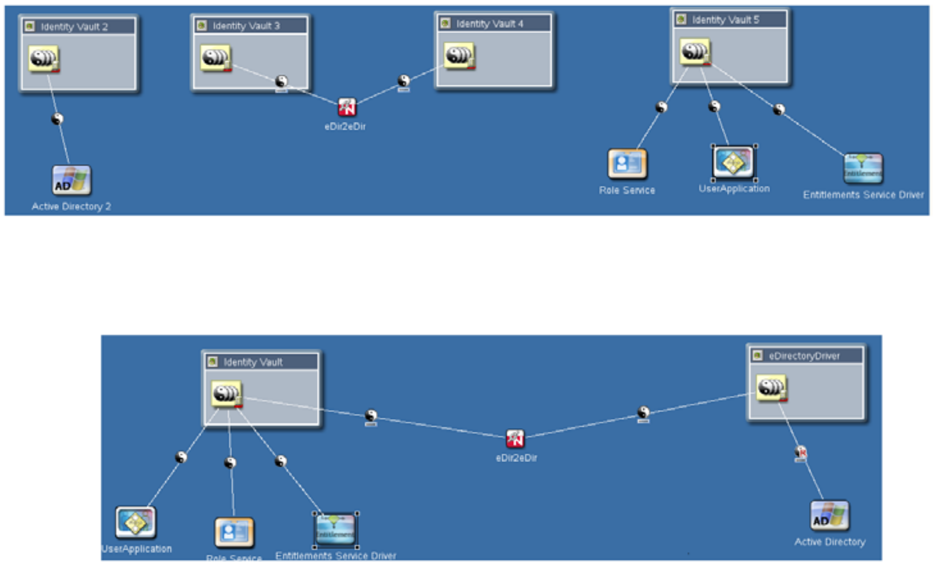


Figure 1-1 shows a basic representation of the movement of Identity Manager projects across different stages. Three projects are developed and tested in three individual setups and then connected in the next stage. The scale of projects grows with stages, but not necessarily the scale of an individual driver.

Managing the movement of a tested solution across different stages and ensuring that nothing is left out can be challenging for Identity Management consultants and customers. When Identity Manager projects are moved from one stage to another, moving driver configurations becomes critical. To successfully stage Identity Manager projects, certain other configurations must also be taken care of. Novell Identity Manager Best Practices Staging Guide provides step-by-step procedures to move your Identity Management solutions from one stage to the subsequent stages. The guide helps you to reduce complexity in your Identity Manager deployment process, by helping you to test your Identity Manager project at multiple stages before it is live.

In the following sections, you will find Designer at the center of discussion. As you already know, Designer is critical to the Identity Manager project development, for developing policies, drivers, PRDs, and so on. Designer is highly capable of bringing/creating the Identity Manager configurations required for running an Identity Manager project and then deploying the configuration on another Identity Manager deployment. You must make certain other changes that are discussed in this guide. Designer can also export Identity Manager environments into a single configuration file and use the file later in a different environment. You can also use any version control system to distribute projects.

Before you begin to use this guide, you should be familiar with Identity Manager and Designer. As you create projects, you should have a uniform Identity Vault design across all the states so that common objects are available. Some objects are moved automatically by Designer, but others should be moved explicitly to make them available in the next stage. See Chapter 2, "Preparing for Staging," on page 11 for more information.

Moving authorizations across stages is a key issue for an Identity Vault security model. eDirectory™ authorizations are assigned to individual objects or to a collection of objects. These authorizations play an important role in the object security because they determine the permission to access the object to which they have been assigned. eDirectory authorizations can be performed through Access Control Lists (ACLs) or Security Equivalances/Exclude Roles. Drivers, jobs, RBEs, and so on should have enough permissions to successfully perform the desired operations. See Chapter 2, "Preparing for Staging," on page 11 for more information.

1.2 Staging Use Cases

The staging discussion is centered on these scenarios:

- **New Deployment:** New drivers and applications are developed during the development stage, then they are tested and moved to the test setup. These applications are put together and moved to the production setup.
- **Existing Deployment:** You already have a development, test, and production setup ready and you want to move a new policy from the development setup in the production setup.

  Existing Deployment discussion is not scoped for the current version of this document.

Preparing for Staging 2

The information covered in the following sections help you design your Identity Manager projects in order to stage them easily. The abilities that Designer provides for staging the projects and some challenges that you might face are also discussed.

- Section 2.1, "Identity Vault Structure," on page 11
- Section 2.2, "Drivers," on page 11
- Section 2.3, "Objects That Designer Models," on page 12
- Section 2.4, "Objects That Designer Does Not Model," on page 13
- Section 2.5, "Rights," on page 14

2.1 Identity Vault Structure

An Identity Vault is primarily a flat eDirectory tree, which consists of several containers for users, devices, groups, objects, and so on. Objects are stored in different containers for performance reasons.

Make sure that you are familiar with the basic principles of directory design. A uniform directory design simplifies administrative tasks for staging. For more information on directory design, refer to the Directory Design for Identity Management Solutions (http://www.novell.com/coolsolutions/appnote/14533.html).

2.2 Drivers

You must create a common data model to allow drivers to work together.

- Section 2.2.1, "Driver Configuration," on page 11
- Section 2.2.2, "Using GCVs in Policies," on page 11
- Section 2.2.3, "Simulation and Staging," on page 12

2.2.1 Driver Configuration

Even though each driver is unique and uses different policies, all drivers use the same guidelines to make the driver configuration file consistent. For example, all policies and driver configuration files have the same naming conventions and support the same common data module.

See Identity Manager Driver Configuration Development Guidelines (http://www.novell.com/documentation/ncmp10/rk12_architecture/data/bg89kav.html) for guidelines on developing new drivers.

2.2.2 Using GCVs in Policies

Global Configuration Values (GCV) are global configuration values or constants, not global variables. There is no way to change a GCV value at runtime. The GCVs are globally accessible to the driver and driver set, but not to the tree or network. GCVs can be consumed by all drivers in a driver set or by all policies in a driver. For more information on using GCVs, see When and How to Use GCVs (http://www.novell.com/documentation/ncmp10/rk12_architecture/data/bg9dfeg.html) in the *Identity Manager Resource Kit 1.2 Architecture Reference Guide*.

2.2.3 Simulation and Staging

The Policy Simulator allows you to test and debug a single policy or a group of policies contained in a policy set or all the policies in a driver or a driver set without implementing the policy in the Identity Vault. It also provides a graphical editor to create the XDS Input documents. You can use these features to test the policies without affecting the production environment or the connected system. This means that you can essentially use Designer as the first stage in your deployment process by developing and then testing your policies through simulation.

2.3 Objects That Designer Models

Designer provides the ability to develop Identity Manager projects even in offline mode. You can easily move your Identity Manager objects from one environment to another. You can also export and import projects into a simple configuration file, which can be stored for future use.

You can model the following objects in Designer:

Feature	Description
Driver Sets	A driver set is a container that holds Identity Manager drivers. Only one driver set can be active on a server at a time. As a result, all active drivers must be grouped into the same driver set.
Drivers	A driver provides the connection between an application and the Identity Vault. The driver is the connector that enables data synchronization and sharing between systems.
GCVs on Driver set and Drivers	Global configuration values (GCVs) are settings that are similar to driver parameters. GCVs can be specified for an individual driver as well as a driver set. If a driver does not have a GCV, the driver inherits the value for that GCV from the driver set.
Policies	Policies cover DirXMLScript, ECMAScript, Entitlement, MappingTable, Resource, Credential Application, Credential Respository, SchemaMap, Filter, and XSLT.
Libraries	You need to provide a context if the library is outside the driver set.
Provisioning Objects	Workflows, roles, resources, teams, etc.
Notification Templates	Notification templates enable you to customize and send e-mail messages that users receive when triggers occur.
Identity Vault Schema, Application Schema	
Role Based Entitlements	Identity Manager allows you to synchronize data between connected systems. Entitlements allow you to set up criteria for a person or group that, once met, initiate an event to grant or revoke access to business resources within the connected system.
Named Passwords	

2.4 Objects That Designer Does Not Model

Feature	Description
O (Organization) and OU (Organizational Unit)	Ensure that O or OU objects are created before deploying them. Import the containers that contain O or OU objects. The following objects must be included in O or OU objects: ◆ All O or OU objects that are Security Equivalences objects for any drivers. ◆ O or OU objects that are used in any policies. ◆ O or OU objects that are used in any job configurations. ◆ O or OU objects that are used in GCVs.
Users	Ensure that the User objects are created before deploying them, especially the admin users. The list of users can be collected in two different ways: Import the containers that contain the user objects. The following objects must be included in the list: ◆ Security Equivalences and Exclude Administrator Roles for all the drivers. ◆ Static Members on groups and RBE policies. ◆ Search identities and Memebrship Filter on Dynamic groups and RBE policies. ◆ Users that are used in any policies. ◆ Users that are used in any job configurations. ◆ Users that are used in GCVs.
Groups	Ensure that the static and dynamic group objects are created before deploying them. Import the containers that contain the groups. The following objects must be included in the list: ◆ Groups that are used in any policies. ◆ Groups that are used in any job configurations. ◆ Groups that are used in GCVs.
Password Policies	Ensure that the policies are created before deploying them.
Indices	Ensure that indices are created before deploying them.
Custom Objects	User-defined objects ar not defined in Designer. Manually create them before deploying. Import the containers that contain the custom objects. The following objects must be included in the list: ◆ All custom objects that are Security Equivalences objects for all the drivers. ◆ Custom objects that are used in any policies. ◆ Custom objects that are used in any job configurations. ◆ Custom objects that are used in GCVs.

Designer 3.5 and later allows you to import objects listed in the above table in LDIF format and then deploy them along with other objects that are being deployed.

NOTE: These objects are not modeled as drivers or driver sets in Designer. They can be modified by modifying the LDIF file that contains these objects in Designer. For more information, refer to Enabling Staging of Projects (http://www.novell.com/documentation/designer35/admin_guide/data/staging_projects.html) in the *Designer 3.5 Administration Guide* (http://www.novell.com/documentation/designer35/index.html).

2.5 Rights

- Section 2.5.1, "Driver Equivalences," on page 14
- Section 2.5.2, "Roles Based Entitlements Policies," on page 14
- Section 2.5.3, "Jobs," on page 15

2.5.1 Driver Equivalences

Security Equivalences require rights to the objects within the Identity Vault in order to perform tasks on them. For example, an Oracle™ database driver has a policy to create a user in the Identity Vault in a container every time a user is created in the database, but the driver doesn't have enough permissions on the container to create the user, so the process fails. The driver has similar rights as that of the users/objects who have permissions on the container. All the policies should be carefully evaluated for finding out what permissions should be given to the drivers.

Designer 3.5 and later can store the Security Equivalences and Exclude Administrative Roles of the drivers in the project and can assign them to the drivers. Before moving to another staging environment, ensure that you know the Security Equivalences and Exclude Administrative Roles associated with each driver and ensure that these objects are imported as LDIF objects and moved along with other objects before being assigned in the next stage after deployment.

If the Security Equivalences object and the Exclude Administrative Roles objects are stored as LDIF objects, Designer ensures that they are created in the next stage before they are assigned.

2.5.2 Roles Based Entitlements Policies

Roles Based Entitlements policies are used by the Entitlements Service driver, which grants entitlements to and revokes entitlements from the users.

An entitlement policy contains the following:

Membership: The list of users assigned to a policy. A user can be dynamically assigned to a policy when he or she meets the criteria for the policy, or the user can be statically (manually) assigned to the policy.

Entitlements: The list of entitlements associated with the policy. Users assigned to the policy receive all of the entitlements associated with the policy. If the user is removed from the policy, he or she loses all entitlements associated with the policy.

You can assign any Identity Vault objects for which you want the entitlement policy to be a trustee. Each member of the policy becomes a trustee of the objects you add.

There are several reasons why you might want to make the policy a trustee of an object:

- One of the policy’s entitlements requires the policy’s members to have rights to an object.
- You want to use the policy to assign users as trustees of an object even though rights to the object are not required for an entitlement. In this case, you are using the entitlement policy to grant and revoke trustee rights for members of the policy.

These rights are not stored in Designer. You should assign the rights after moving to the next stage.

2.5.3 Jobs

Identity Manager has a job scheduling utility that schedules events, such as setting the system to disable an account on a specific day, or initiating a workflow to request an extension for a person to access a corporate resource. The Job Manager runs on every Identity Manager server in the background. Based on the job definition, it checks every minute to see if a job needs to run. When it encounters a job, it runs the appropriate Job implementation.

The Job Manager needs appropriate permissions to run succesfully. For example, a job that disables a user account from the Identity Vault needs adequate permissions. Appropriate access must be granted to the job object in the Identity Vault so that it can modify a user object. Use iManager to grant the required rights for the jobs because Designer does not allow you to grant rights for jobs.

Staging a Project 3

This section contains the following information:

- Section 3.1, "Prerequisites," on page 17
- Section 3.2, "Staging," on page 19
- Section 3.3, "Post-staging," on page 22

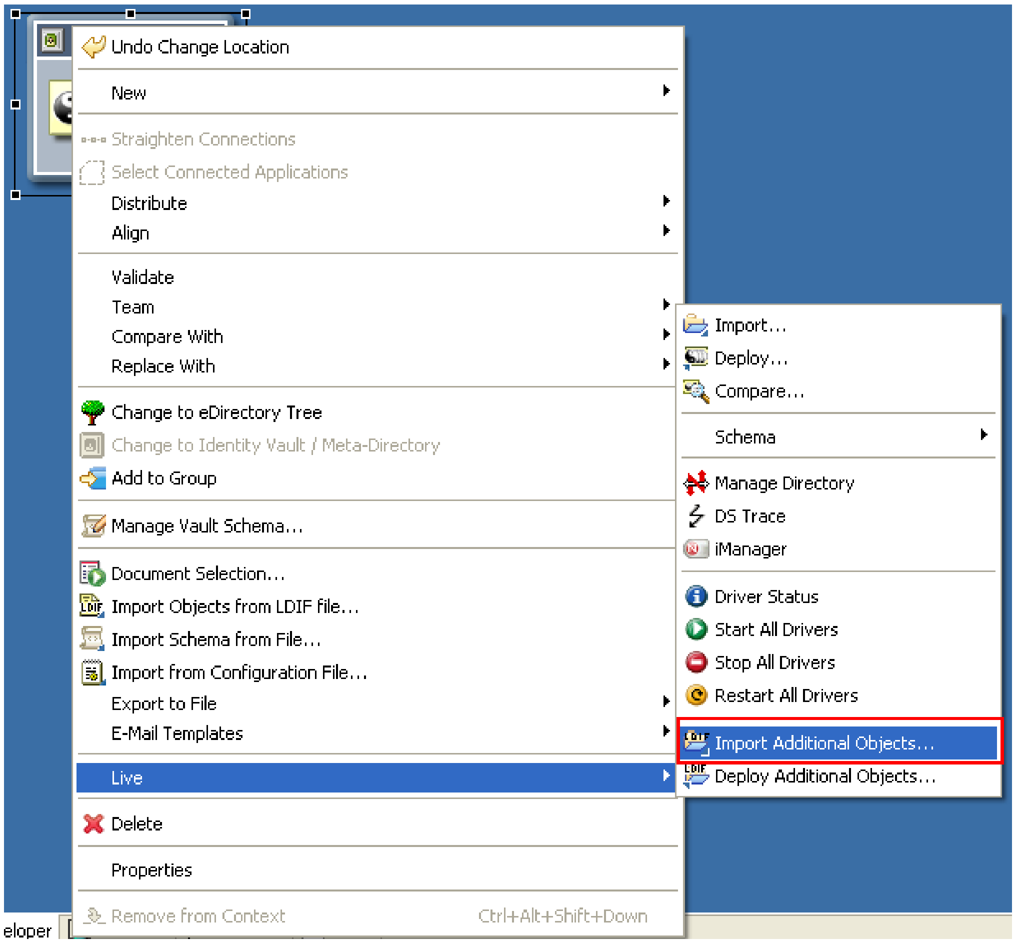
3.1 Prerequisites

Ensure that the following general prerequisites are met before attempting the staging:

- All the stages have the same version of eDirectory, Identity Manager, and Identity Manager drivers.
- Designer 3.5 or later is present.
- Same workspace is used for all the stages.
- All the applications and drivers are fully developed and tested in one stage before moving them to the next stage.
- From your project, gather information about the objects that are not modeled by Designer. For more information, see Section 2.4, "Objects That Designer Does Not Model," on page 13.
- Create an LDIF file for all the objects that are not modeled by Designer. Use Designer to import the additional objects.

3.1.1 Importing Objects

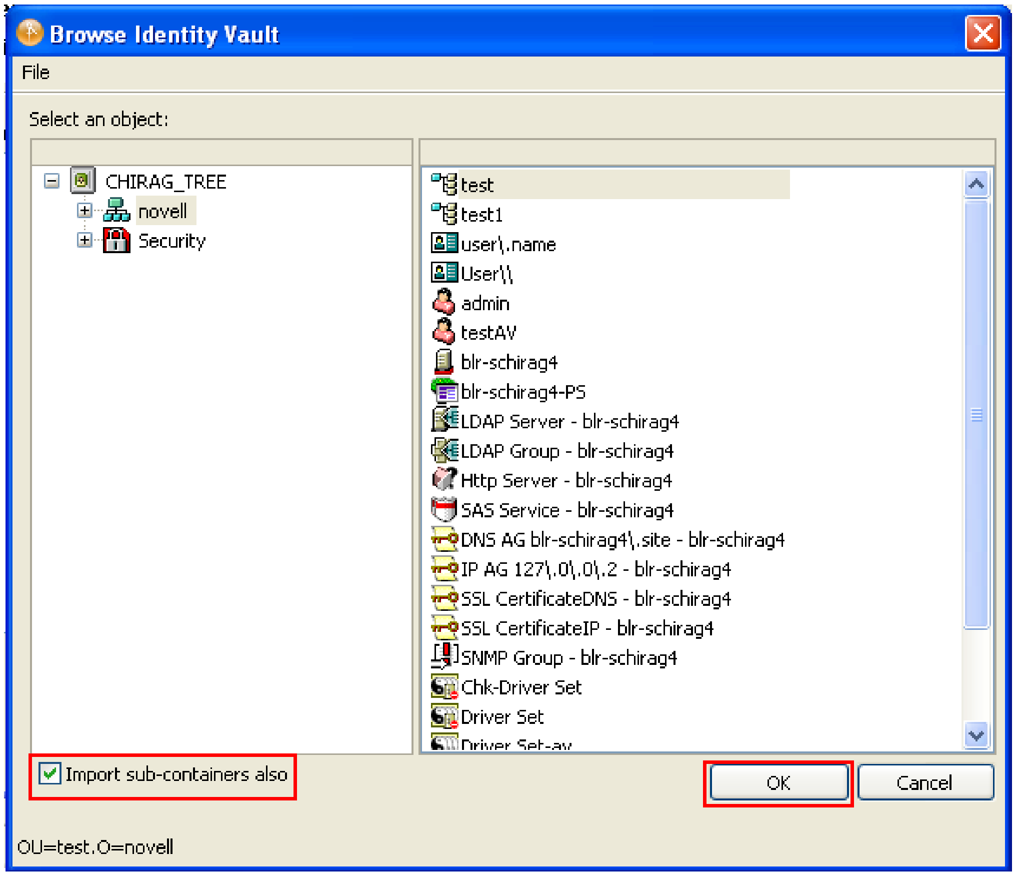
1 In Designer, right-click *Identity Vault* and select *Live* > *Import Additional Objects*.



2 Browse to and select the objects you want to add to the LDIF file.

Or

If you want to select all the objects in a container, select *Import sub-containers also* in the Browse Identity Vault dialog box.


Browse Identity Vault
File
Select an object:
CHIRAG_TREE
novell
Security
test
test1
user\.name
User\\
admin
testAV
blr-schirag4
blr-schirag4-PS
LDAP Server - blr-schirag4
LDAP Group - blr-schirag4
Http Server - blr-schirag4
SAS Service - blr-schirag4
DNS AG blr-schirag4\.site - blr-schirag4
IP AG 127\.0\.0\.2 - blr-schirag4
SSL CertificateDNS - blr-schirag4
SSL CertificateIP - blr-schirag4
SNMP Group - blr-schirag4
Chk-Driver Set
Driver Set
Driver Set-av
Import sub-containers also
OK
Cancel
OU=test.O=novell

3 Click *Continue* on the Import Dialog to import all the objects into Designer.

4 Repeat Step 1 through Step 3 for all the Identity Vaults in your projects.

You can edit the LDIF objects from the LDIF container. Go to the Outline View, expand the Identity Vault, then double-click the LDIF container.

IMPORTANT: You should back up your project by using a version control system or export it to a file.

3.2 Staging

You should ensure that all the applications and Identity Manager systems are up and running in the next stage before moving the configurations.

1 Back up the current stage project to reuse it in the next stage:

1a In Designer, go to *Windows >Show View > Project*.

1b Right-click the current stage project and select *Copy Project*.

1c Enter the name for the next stage project, then click *OK*.

1d Click *Finish*.

2 To use the first stage project in the subsequent stages, you should rename it.

- **2a** In the project view, right-click the first stage project and select *Rename*.
- **2b** In Designer, click *File >Import* to import the project exported in Step 1.
- **2c** In the Import window, select *Designer for Identity Manager > Project* (From the Filesystem), then click *Next*.

3 Import the project exported in Step 1.

- **3a** In the Import window, select *Designer for Identity Manager > Project* (From the Filesystem), then click *Next*.
- **3b** In the Import Existing Projects page, select the root directory or archive file and browse to the first stage project.
- **3c** Click *Finish*.

4 Change the configuration of one of the Identity Vaults in your project.

- **4a** In the Outline view or the Modeler, double-click the *ID Vault*.
- **4b** In the Configuration page, change the *Hostname, Admin Username,* and *Admin Password*.
- **4c** Click *Test Connection* for checking the connectivity.
- **4d** Under each Identity Vault, double-click the servers and change the name and context of the servers.
- **4e** Add more servers and associate them to the driver set, if necessary.

5 Change the configuration of all the connected systems of this Identity Vault.

- **5a** In the Modeler, double-click a driver or a driver line.
- **5b** In the Driver properties page, change the authentication information in the *Authentication* tab.
- **5c** In the Driver properties page, change the driver related information in the *Driver Parameters* tab.
- **5d** The driver parameters depend on the servers on which the drivers reside. Ensure that you change the driver parameters on multiple servers if you have multiple servers running a driver.

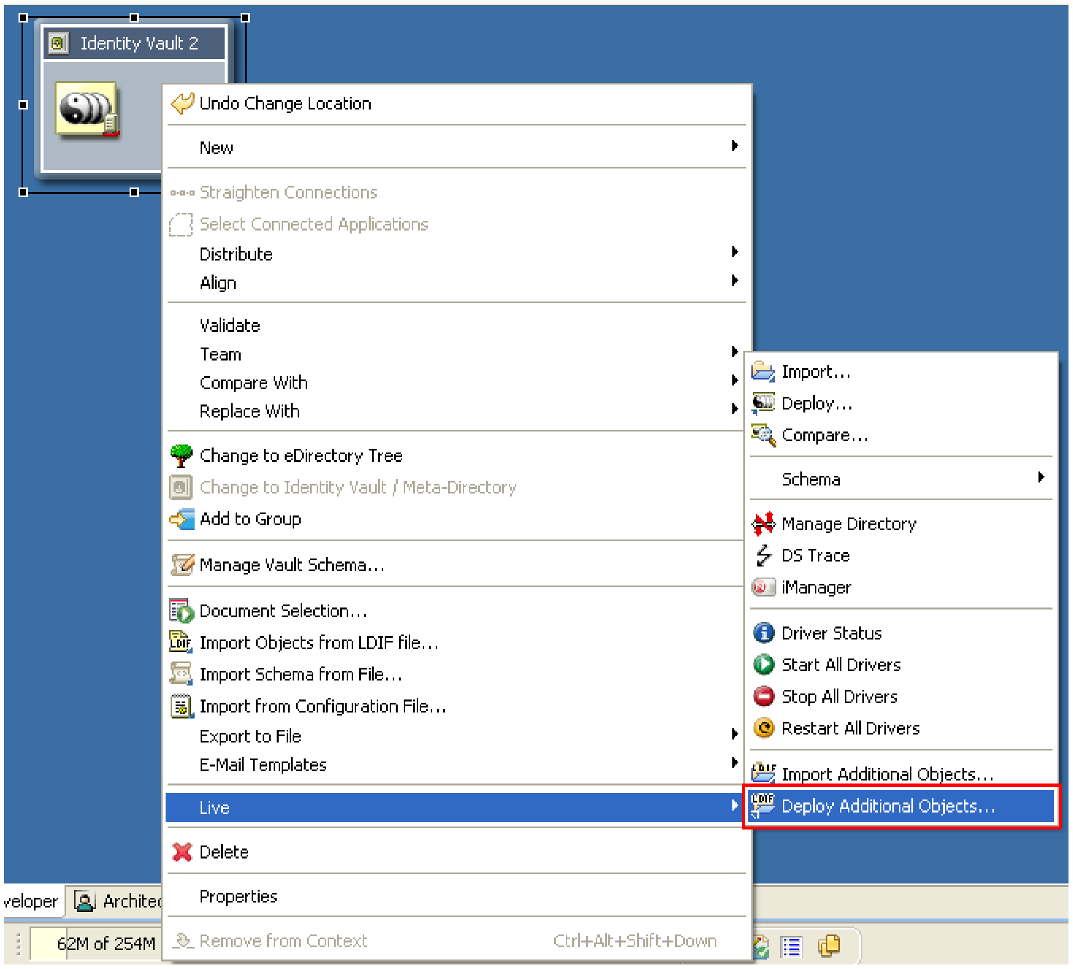
6 Change the GCVs for all drivers and driver sets of this Identity Vault.

GCVs should be the only changes that you make on the drivers and the driver set along with the configuration. Your policies won't change if they are properly designed.

- **6a** Monitor all the GCVs that should be changed with the movement of projects.
- **6b** Change all the GCVs that change with the environment.
- **6c** Move or add new GCVs to the new servers added in Step 4e.

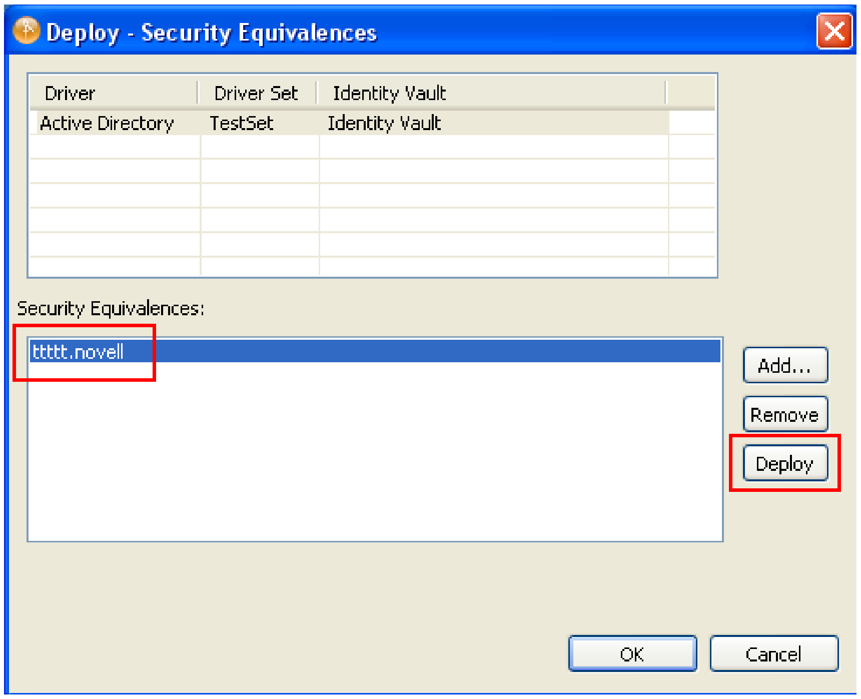
7 Run the project checker to ensure integrity of your project. To deploy the schema, right-click *ID Vault > Live > Schema > Deploy*.

8 To deploy additional objects gathered in the Prerequisites, right-click *ID Vault > Live > Deploy Additional Objects*.



9 To deploy the Identity Vault, right-click *ID Vault* > *Live* > *Deploy.*

10 Deploy the appropriate Security Equivalences and Exclude Admin Roles objects for each driver. See Section 2.5, “Rights,” on page 14 for more information.



Driver	Driver Set	Identity Vault
Active Directory	TestSet	Identity Vault

Security Equivalences:

ttttt.novell

11 Repeat Step 4 through Step 10 for each Identity Vault in your project.

3.3 Post-staging

Designer does not move all the configurations to the next stage. Users are expected to manually perform a few tasks to ensure that the configurations work properly.

- **Security Equivalences and Exclude Admin Roles:** Check whether all the drivers have appropriate Security Equivalences and Exclude Admin Roles objects as defined in the previous stage.
- **eDir2eDir Driver Certificates:** If you have *eDir2eDir* driver certificates created in the current stage, ensure that these certificates are created in the next stage.
  1. In Designer, right-click the *eDir2eDir* application, then click *Secure Connection Settings*.
  2. Click *Enable SSl/TLS*, select the required options, then click *OK*.
  3. Right-click *eDir2eDir*, then click *Live >Create eDir-to-eDir Certificates*.
- **Java Environment Parameters:** The Java* environment parameters enable you to configure the Java Virtual Machine™ (JVM) on the Metadirectory server associated with the driver set. You might need to change the Java classpath options if the .jar files your Metadirectory server is looking for reside at a different place in the new stage. To change the location, go to *DriverSet Properties Page > Java > ClassPath Additions* and provide the correct classpaths. When you enter multiple class paths, separate them with a semi colon (;) for a Windows JVM and a colon (:) for a UNIX* or Linux* JVM. Deploy the driver set if you make any changes.
- **Indexes:** Make sure that all the customized indexes from the previous stage have been copied to the new stage. eDirectory uses these indexes to significantly improve the query performance. Some indexes are shipped with eDirectory. These default indexes are for the following attributes:
  - CN

    - Aliased Object Name
    - dc
    - Obituary
    - Given Name
    - Member
    - Surname
    - Reference
    - uniqueID
    - Equivalent to Me
    - GUID
    - NLS: Common Certificate
    - cn_SS
    - Revision
    - uniqueID_SS
    - extensionInfo
    - ldapAttributeList
    - ldapClassList

  You can visit each Identity Vault server and collect the customized index information by doing the following:

    1. In Novell® iManager, click the *Roles and Tasks* tab.
    2. Click *eDirectory Maintenance > Index Management*.
    3. Select a server from the list of available servers.

       It lists all the active and offline indexes on the selected server.
    4. Make a note of all the customized indexes.

       Ensure that you add these indexes to the corresponding servers in the next stage. See Index Manager (http://www.novell.com/documentation/edir88/edir88/data/a5tuuu5.html) for more information on creating, adding, or deleting indexes.

- Ensure that password policies assigned to the containers, users, groups in the previous stage are assigned again in the current stage.
- Start the drivers after moving the driver configuration to the next stage. Right-click each driver, select *Live*, then select *Start Driver*.

4 Best Practices in Moving Objects Across Stages

- If you delete drivers and driver sets from Stage 2 in order to deploy the drivers from Stage 1, you can lose the associations.
- Don't deploy the Stage 1 objects directly into the Stage 2 environment.

  Always use the configuration file, the exported project archive files, and the LDIF files of the Stage 1 setup.
- Before moving to any stage, understand the existing stage and the objects that Designer does not automatically bring in (see "Objects That Designer Does Not Model" on page 13) for the next stage.

  Ensure that you know which objects are required in the subsequent stages. Consolidate these objects in the LDIF file.
- Ensure that you assign the Security Equivalences, Trustees, and Server Certificates of Stage 1 in Stage 2 after deployment.
- LDIF files that contain additional objects should be stored locally. You can use Import Convert Export (ICE) utility to deploy these objects in any stage.
- For a new deployment in Stage 2, ensure that LDIF objects are deployed before importing the configuration file or the project file.
- For an existing deployment in Stage 2, ensure that you compare the existing project with the Stage 1 configuration, deploy the necessary LDIF objects, then import the configuration file.
- Ensure that objects are up-to-date when you import them into the LDIF file.

  Always import the additional objects into Stage 1 before moving to Stage 2.
- Export the additional objects of Stage 1 into an LDIF file before moving to Stage 2 so that these objects can be manually created in Stage 2 before deployment.