EVIDEN

Identity and Access Management

DirX Identity

Migration Guide

Version 8.10.17, Edition November 2026



All product names quoted are trademarks or registered trademarks of the manufacturers concerned.

© 2026 Atos Group

All Rights Reserved

Distribution and reproduction not permitted without the consent of Atos Group.

Table of Contents

Copyright ii
Preface 1
DirX Identity Documentation Set 2
Notation Conventions 4
1. Introduction 6
1.1. Migration from Versions older than V8.3 6
1.2. Migration from Versions older than V8.7 6
1.3. Manual Migration Overview 6
1.3.1. Web Center Migration Overview 7
1.4. Features to be Migrated 7
1.4.1. General Issues 7
1.4.1.1. Deletion of Old Objects 7
1.4.2. Issues Relevant for Upgrade from V8.2A 8
1.4.2.1. Typo ClearOnMasterRemoval 8
1.4.2.2. Identity API 8
1.4.2.3. Adaption of Policies 8
1.4.3. Issues Relevant for Upgrade from V8.2B 8
1.4.3.1. Migration of JMS Subscriptions 8
1.4.3.2. Migration of Activation Date in Request Workflow Instances 9
1.4.3.3. Migration of XML Content to LDAP Attributes in Request Workflow/Activity instances 9
1.4.4. Issues Relevant for Upgrade from V8.2C 9
1.4.4.1. Migration of Realtime Channels to Support Delta Handling 9
1.4.4.2. Migration of Event Port in Realtime Workflows 9
1.4.4.3. Deletion of the Certificate Handler adaptor in the IdS-J 10
1.4.4.4. Issues Relevant for Upgrade from V8.3 or V8.3 R2 10
1.4.4.5. JMS Audit Handler 10
1.4.5. Issues Relevant for Upgrade from V8.4 10
1.4.5.1. Java-based Server 10
1.4.6. Issues Relevant for Upgrade from V8.5 10
1.4.6.1. Use of SSL 10
1.4.7. Issues Relevant for Upgrade from V8.6 11
1.4.7.1. Clear Text Passwords 11
1.4.8. Issues Relevant for Upgrade from V8.7 11
1.4.8.1. Topic Prefixes 11
1.4.9. Issues Relevant for Upgrade from V8.9 11
1.4.9.1. Client-side SSL Support in Realtime Workflows 11
1.4.9.2. New LDAP Controls LDAP_CTRL_NO_MODTIME_UPD (1.3.12.2.1107.1.3.2.12.9) and LDAP Conditioned Operation Control (1.3.12.2.1107.1.3.2.12.10) 12

1.4.9.3. Topic Prefixes . . . 12
2. Migration Procedure . . . 13
3. Preparing the Migration . . . 14
3.1. Preserving Files . . . 14
3.2. Customized Web Center . . . 14
3.3. Provisioning Web Services and Related Clients . . . 14
3.4. Update Tomcat and Java Environment . . . 15
3.5. Service Configuration Cleanup Linux . . . 15
4. Automatic Migration during Configuration . . . 16
5. Manual Migration . . . 17
5.1. General Aspects . . . 17
5.1.1. Restoring Preserved Files . . . 17
5.1.2. Migrating Passwords for JMS-based Auditing . . . 17
5.1.3. Migrating Passwords for Mail and SMS Gateway Configuration . . . 17
5.1.4. Migrating Passwords for Sending Notifications in Tcl-based Workflows . . . 18
5.1.5. Using SAP JCo version 3.1.4 or Higher for SAP ECC UM Connector/Agent . . . 18
5.1.6. Migrating SPML Provisioning Web Services . . . 18
5.1.7. Migrating SPML Provisioning Web Services Clients . . . 18
5.1.8. Migrating a Customized Web Center . . . 19
5.1.9. Migrating a Customized Web Center for SAP NetWeaver . . . 19
5.1.10. Migrating a Customized Web Center for Password Management . . . 20
5.1.11. Migrating a Customized Business User Interface . . . 20
5.1.12. Migrating the ActiveMQ Messaging Server . . . 20
5.1.13. Adapting Size Limits for LDAP Searches (using DirX Directory) . . . 20
5.1.14. Extending the DirX Identity Schema for Customer Domains . . . 22
5.1.14.1. Agent Schema Changes in V8.10 . . . 22
5.1.14.2. Agent Schema Changes in V8.9 . . . 22
5.1.14.3. Agent Schema Changes in V8.7 . . . 22
5.1.14.4. Agent Schema Changes in V8.6 . . . 22
5.1.14.4.1. RACF . . . 22
5.1.14.4.2. Salesforce . . . 24
5.1.14.5. Agent Schema Changes in V8.5 . . . 25
5.1.14.5.1. ADS . . . 25
5.1.14.6. Agent Schema Changes in V8.4 . . . 25
5.1.15. LDAP Lock Mechanism for User Objects . . . 25
5.1.16. Migrating Realtime Workflows to Support Client-side SSL Authentication . . . 25
5.2. Aspects Relevant to Using DirX Directory V8.6 or Higher . . . 26
5.2.1. Rebuilding Attribute Indexes to Handle Improved Search Operations . . . 26
5.2.2. Migration of New LDAP Controls LDAP_CTRL_NO_MODTIME_UPD (1.3.12.2.1107.1.3.2.12.9) and LDAP Conditioned Operation Control (1.3.12.2.1107.1.3.2.12.10) . . . 27
5.3. Aspects Relevant to DirX Identity V8.10 . . . 27

5.3.1. Reports based on TIBCO Jaspersoft . . . 27
5.3.2. Support of Java 11 and Tomcat 9 . . . 27
5.4. Aspects Relevant to DirX Identity V8.9 . . . 27
5.4.1. Support of Java SE 11 . . . 28
5.4.2. Support of Apache Tomcat 9 . . . 28
5.4.3. JMX Access . . . 28
5.4.4. Class Loading in IdS-J Server (for Customer-specific Request Workflow Job Implementations) . . . 28
5.4.5. Handling of Escaping in Provisioning Rule Filters (RJYNS3) . . . 30
5.4.6. Old and New Delegations . . . 30
5.4.6.1. Old Delegations . . . 30
5.4.6.2. New Delegations . . . 31
5.4.6.3. Switching Between Old and New Delegations . . . 31
5.4.6.4. Access Policy Evaluation . . . 32
5.4.6.5. Migrating to DirX Identity 8.9 . . . 32
5.5. Aspects Relevant for Upgrade from V8.6 . . . 32
5.5.1. Replacement of REST Approval Service with REST Service . . . 32
5.5.2. Replacement of HTML5-based Approval App with Business User Interface App . . . 32
5.5.3. Migrating Identity Server SSL Configurations . . . 32
5.5.3.1. Preparing Existing Key Material . . . 33
5.5.3.2. Generating New Client Key Material . . . 33
5.5.3.3. Distributing Shared Key Material . . . 34
5.5.4. Migrating Windows Password Listeners Connecting with SSL to ActiveMQ to Connect with Client-side SSL . . . 34
5.5.5. Migrating Passwords for Web Admin and Active MQ Web Console . . . 35
5.6. Aspects Relevant for Upgrade from V8.5 . . . 37
5.7. Aspects Relevant for Upgrade from V8.4 . . . 37
5.8. Aspects Relevant for Upgrade from V8.3 . . . 37
5.8.1. Migrating External Messaging Client Tools . . . 37
5.8.2. Updating the JMS Audit Handler Deployment . . . 37
5.8.3. Displaying Inherited Privileges . . . 37
5.8.4. Filtering Assigned Privileges . . . 38
5.8.5. Migrating Role Parameter Access and Data . . . 38
5.8.5.1. Migrating Java Code for Accessing Role Parameter Values . . . 38
5.8.5.2. Migrating Role Parameter Data in Assignments . . . 40
5.8.5.3. Migrating Role Parameter Data in Running Request Workflows . . . 40
5.8.6. Migrating Tcl-based Workflows to Handle Changes in the Object Search Operation . . . 40
5.8.7. Migrating LDAP Realtime Workflows for Group Renaming . . . 41
5.8.8. Migrating Tcl-based and Realtime Workflows for New Object Classes dxrPersona, dxrFunctionalUser and dxrUserFacet . . . 42

5.8.8.1. Adapting Tcl (Import) Workflows . . . 42
5.8.8.2. Adapting Tcl (Export) Workflows . . . 43
5.8.8.3. Adapting Realtime Workflows . . . 43
5.8.9. Migrating the JMS Audit Configuration . . . 45
5.9. Aspects Relevant for Upgrade from V8.2 . . . 45
5.9.1. Migrating Identity Server SSL Configurations . . . 45
5.9.1.1. Using the Java-based Server Key Material . . . 45
5.9.1.2. Using the Messaging Service Key Material . . . 46
5.9.1.3. Using the Key Material from the C++-based Server SOAP Port . . . 47
5.9.2. Cleanup regarding Worker Containers . . . 48
5.9.2.1. Windows Platforms . . . 49
5.9.2.2. Linux Platforms . . . 49
5.9.3. Cleanup regarding JAXB-API and JAXWS-API in Tomcat . . . 49
5.9.4. Adapting Custom Scripts . . . 49
5.9.4.1. Adapting Java Calls in Windows Batch Files . . . 50
5.9.4.2. Adapting keytool Calls in Windows Batch Files . . . 50
5.9.4.3. Adapting Java Calls in UNIX Shell Scripts . . . 50
5.9.4.4. Adapting keytool Calls in UNIX Shell Scripts . . . 50
5.9.4.5. Adapting Java Calls in Tcl scripts . . . 51
5.10. Aspects Relevant for Upgrade from V8.2C . . . 51
5.10.1. Deleting the jms.jar File . . . 51
5.10.2. Migrating Realtime Channels to Support Realtime Delta Workflows . . . 51
5.10.3. Updating the Realtime Event Port . . . 52
5.10.4. Handling Minimum Source Entries in Tcl-based Workflows . . . 52
5.10.5. Creating the dxrUid Attribute in the Provisioning Tree . . . 52
5.10.6. Migrating the e-Mail Body in Reject e-Mails of Request Workflows . . . 53
5.10.7. Migrating the URL of the Source Ticketing Sample Web Service . . . 54
5.10.8. Migrating SPML Filters that Use Wildcards . . . 54
5.11. Aspects Relevant for Upgrade from V8.2B . . . 55
5.11.1. Updating Manager Profiles for the Data View . . . 55
5.11.2. Migrating JMS Subscriptions . . . 56
5.11.3. Migrating the Source Ticketing Sample Web Service . . . 56
5.11.4. Migrating Request Workflow Parameters to be Stored as dxmSpecificAttributes . . . 56
5.12. Aspects Relevant for Upgrade from V8.2A . . . 57
5.12.1. Adapting Object Descriptions . . . 57
5.12.2. Updating the SSL Flag for Messaging . . . 57
5.12.3. Solving Class Loading Problems in IdS-J Server . . . 57
5.12.4. Adjusting SAP R/3 UM Workflows . . . 58
5.12.5. Fixing the ClearOnMasterRemoval Typo . . . 58
5.12.6. Using the New Identity API . . . 59
5.12.7. Adapting Policies . . . 59

5.12.8. Migrating the Source Ticketing Sample Web Service . . . 59
6. Known Issues . . . 60
6.1. Overwritten cert8.db file during Update installation . . . 60
6.2. Overwritten agent batch files during Update installation . . . 60
6.3. Deletion of Old Objects . . . 60
6.4. UID Generation Fails . . . 61
6.5. Sample Domain: Doubled Memberships (ADD/OK) . . . 61
6.6. Inconsistent Object Descriptions (single / multi value) . . . 62
7. Migrating JavaScript from Rhino to GraalVM . . . 63
7.1. Before You Migrate . . . 63
7.2. Selecting the JavaScript Executor . . . 63
7.3. Adapting JavaScript Code . . . 64
7.3.1. Import Java Classes Explicitly . . . 64
7.3.2. Check Strings, Java Calls, and Result Types . . . 64
7.4. Rolling Out and Troubleshooting . . . 65
Further Useful Links for GraalVM Migration . . . 65
Legal Remarks . . . 67

Preface

The *DirX Identity Migration Guide* is designed to help you to perform the migration from previous versions. It consists of the following sections:

- Chapter 1 discusses the migration process and describes the concepts.
- Chapter 2 describes the general migration procedure.
- Chapter 3 describes what must be done before you start migration.
- Chapter 4 describes the automatic part of the migration process.
- Chapter 5 describes how to perform additional manual migration tasks that are required to complete the migration.
- Chapter 6 describes how to solve known post-migration issues.
- Chapter 7 describes how to migrate JavaScript from Rhino to GraalVM and select the executor for each script.

Perfect migration is difficult to achieve because the variety of configuration databases at the customer site is infinite. Thus, you should carefully follow the sequence of steps described in this guide to achieve successful migration. Test all of your workflows and applications after migration to ensure that everything works properly.

DirX Identity Documentation Set

Version 8.10.17 | Build 2130 | Date 2026-11-20

The DirX Identity document set consists of the following manuals:

- *DirX Identity Introduction*. Use this book to obtain a description of DirX Identity architecture and components.
- *DirX Identity Release Notes*. Use this book to understand the features and limitations of the current release. This document is shipped with the DirX Identity installation as the file **release-notes.pdf**.
- *DirX Identity History of Changes*. Use this book to understand the features of previous releases. This document is shipped with the DirX Identity installation as the file **history-of-changes.pdf**.
- *DirX Identity Tutorial*. Use this book to get familiar quickly with your DirX Identity installation.
- *DirX Identity Provisioning Administration Guide*. Use this book to obtain a description of DirX Identity provisioning architecture and components and to understand the basic tasks of DirX Identity provisioning administration using DirX Identity Manager.
- *DirX Identity Connectivity Administration Guide*. Use this book to obtain a description of DirX Identity connectivity architecture and components and to understand the basic tasks of DirX Identity connectivity administration using DirX Identity Manager.
- *DirX Identity User Interfaces Guide*. Use this book to obtain a description of the user interfaces provided with DirX Identity.
- *DirX Identity Application Development Guide*. Use this book to obtain information how to extend DirX Identity and to use the default applications.
- *DirX Identity Customization Guide*. Use this book to customize your DirX Identity environment.
- *DirX Identity Integration Framework*. Use this book to understand the DirX Identity framework and to obtain a description how to extend DirX Identity.
- *DirX Identity Web Center Reference*. Use this book to obtain reference information about the DirX Identity Web Center.
- *DirX Identity Web Center Customization Guide*. Use this book to obtain information how to customize the DirX Identity Web Center.
- *DirX Identity Meta Controller Reference*. Use this book to obtain reference information about the DirX Identity meta controller and its associated command-line programs and files.
- *DirX Identity Connectivity Reference*. Use this book to obtain reference information about the DirX Identity agent programs, scripts, and files.
- *DirX Identity Troubleshooting Guide*. Use this book to track down and solve problems in your DirX Identity installation.
- *DirX Identity Installation Guide*. Use this book to install DirX Identity.

- *DirX Identity Migration Guide*. Use this book to migrate from previous versions.

Notation Conventions

Boldface type
In command syntax, bold words and characters represent commands or keywords that must be entered exactly as shown.

In examples, bold words and characters represent user input.

Italic type
In command syntax, italic words and characters represent placeholders for information that you must supply.

[ ]
In command syntax, square braces enclose optional items.

{ }
In command syntax, braces enclose a list from which you must choose one item.

In Tcl syntax, you must actually type in the braces, which will appear in boldface type.

|
In command syntax, the vertical bar separates items in a list of choices.

...
In command syntax, ellipses indicate that the previous item can be repeated.

userID_home_directory
The exact name of the home directory. The default home directory is the home directory of the specified UNIX user, who is logged in on UNIX systems. In this manual, the home pathname is represented by the notation *userID_home_directory*.

install_path
The exact name of the root of the directory where DirX Identity programs and files are installed. The default installation directory is *userID_home_directory*/**DirX Identity** on UNIX systems and **C:\Program Files\DirX\Identity** on Windows systems. During installation the installation directory can be specified. In this manual, the installation-specific portion of pathnames is represented by the notation *install_path*.

dirx_install_path
The exact name of the root of the directory where DirX programs and files are installed. The default installation directory is *userID_home_directory*/**DirX** on UNIX systems and **C:\Program Files\DirX** on Windows systems. During installation the installation directory can be specified. In this manual, the installation-specific portion of pathname is represented by the notation *dirx_install_path*.

dxi_java_home
The exact name of the root directory of the Java environment for DirX Identity. This location is specified while installing the product. For details see the sections "Installation" and "The Java for DirX Identity".

tmp_path

The exact name of the tmp directory. The default tmp directory is /tmp on UNIX systems. In this manual, the tmp pathname is represented by the notation *tmp_path*.

tomcat_install_path
The exact name of the root of the directory where Apache Tomcat programs and files are installed. This location is defined during product installation.

mount_point
The mount point for DVD device (for example, **/cdrom/cdrom0**).

1. Introduction

New customers install DirX Identity and work with the product. Existing customers must migrate their environment.

1.1. Migration from Versions older than V8.3

With DirX Identity V8.3, the Messaging functionality was changed.If you want to preserve and process your unprocessed messages (like password changes, Provisioning events, and so on), you must migrate them to the new Messaging functionality.DirX Identity V8.3 offers a migration tool for this task.In this case, you must migrate to V8.3. After successful migration to V8.3, you can migrate to the current version in a second step.See the V8.3 migration guide for information on how to migrate to V8.3. If you are not interested in message migration, you can migrate directly to V8.6.

1.2. Migration from Versions older than V8.7

This section is only of interest if you use the RACF schema extensions.

In all versions older than V8.7, the following RACF attributes were incorrectly defined with conflicting equality matching rules (Case Ignore equality matching) and substring matching rules (Case Exact substring matching).

- racfLNotesShortName
- racfNdsUserName
- krbPrincipalName
- racfLdapBindDN
- racfLdapBindPw

To solve this problem, we provide an automatic migration procedure that runs as part of the DirX Identity Configuration tool.The procedure copies the attribute values from the erroneous attributes to the new attributes that are correctly defined.We have renamed the erroneous attributes to LDAP attribute names with the suffix **OLD**; for example, **racfLNotesShortNameOLD**.After the data has been migrated in LDAP, you'll need to update the LDAP schema to remove these renamed erroneous attributes from several object classes.For details, see the section "Extending the DirX Identity Schema for Customer Domains" in the chapter "Manual Migration".

1.3. Manual Migration Overview

The following table lists the relevant manual steps described in the sections of chapter "Manual Migration" you should consider depending on your base version:

Base \ Target version \ version	**V8.10**
V8.9	"General Aspects" and "Aspects Relevant for DirX Identity V8.9"

Base \ Target version \ version	V8.10
V8.7	Additional to the sections above “Aspects Relevant for Upgrade from V8.7” in this chapter
V8.6	Additional to the sections above “Aspects Relevant for Upgrade from V8.6” in this chapter
V8.5	Additional to the sections above “Aspects Relevant for Upgrade from V8.5” in this chapter
V8.4	Additional to the sections above “Aspects Relevant for Upgrade from V8.4” in this chapter
V8.3 , V8.3 R2	Additional to the sections above “Aspects Relevant for Upgrade from V8.3” in this chapter
V8.2C	Additional to the sections above “Aspects Relevant for Upgrade from V8.2C” in this chapter
V8.2B	Additional to the sections above “Aspects Relevant for Upgrade from V8.2B” in this chapter
V8.2A	All sections.

1.3.1. Web Center Migration Overview

For Web Center migration, you need to consider all the change files on the version path.For example, if you want to migrate from V8.2A SP2 to V8.5, you need to read the following files:

- WebCenterChanges-82A-SP2-to-82C.txt
- WebCenterChanges-82C-to-83.txt
- WebCenterChanges-83-to-84.txt
- WebCenterChanges-83-to-85.txt

1.4. Features to be Migrated

This section gives an overview of the features to be migrated and explains the underlying concepts.

Many items in the next sections are performed automatically, but in some cases, manual steps are necessary.Consider all these next sections accordingly depending on the base version from which you want to upgrade.

1.4.1. General Issues

This section discusses issues that are relevant to all versions.

1.4.1.1. Deletion of Old Objects

Renaming of objects in the Provisioning configuration database requires deletion of the old

objects after adding the new ones. This comes from the fact that the cn, which is part of the DN, is used as a display attribute. Procedures exist that handle the deletion task automatically.

1.4.2. Issues Relevant for Upgrade from V8.2A

This chapter describes all issues that are relevant for V8.2A only.

1.4.2.1. Typo ClearOnMasterRemoval

In the **UserCommon.xml**, this flag was not correctly written.

1.4.2.2. Identity API

The Identity API has changed. For details, see the DirX Identity API documentation delivered with Web Center.

1.4.2.3. Adaption of Policies

The object description names for password policies and provisioning rules had to be changed.

1.4.3. Issues Relevant for Upgrade from V8.2B

This section comprises all issues relevant for V8.2B only.

1.4.3.1. Migration of JMS Subscriptions

Subscription IDs of the C++-based Server were unified. The structure of the subscription IDs is now identical for Windows and UNIX systems. Statustracker subscription IDs no longer include the hostname of the system where it runs, because it may move to another host.

The new format for subscription ID is:

*host***.jms.***module.topic*

For statustracker:

jms.*module.topic*

The old format is:

- Windows:

*host***.jms.***module.host.topic*

For statustracker:

jms.*module.host.topic*

- UNIX:

host.**libmqjms_dxm.***host.topic*

For statustracker:

libmqjms_dxm.*module.host.topic*

An automatic migration routine performs this task during the upgrade configuration when you start the C++-based Server hosting ATS the first time.

1.4.3.2. Migration of Activation Date in Request Workflow Instances

Request workflows started with older versions may include an activation date (the workflow creation date) in the order. This causes a ticket creation during processing of the order. Migration eliminates these activation dates.

1.4.3.3. Migration of XML Content to LDAP Attributes in Request Workflow/Activity instances

In older versions, a lot of attributes were managed by an XML structure in the dxmContent LDAP attribute. For performance reasons, these attributes are now represented by LDAP attributes. Without a migration Request Workflow/Activity instances cannot be displayed correctly in the Identity Manager (especially the graphic). Therefore Request Workflow/Activity instances created by older versions are migrated.

1.4.4. Issues Relevant for Upgrade from V8.2C

This section describes all issues relevant for version V8.2C only.

1.4.4.1. Migration of Realtime Channels to Support Delta Handling

DirX Identity V8.3 supports delta handling in realtime workflows. This new feature requires an extension in the XML definition of the realtime channels. (LDAP attribute “dxmContent”).

An automatic migration routine performs this task during the upgrade configuration.

You can run this routine by hand at any time.

1.4.4.2. Migration of Event Port in Realtime Workflows

The realtime workflows can send JMS events for the event-based workflows. Internally the workflows have an event port integrated that starts a JMS connector. The XML definition for this JMS connector (LDAP attribute “dxmContent”) had some connection parameters defined that were not used by the workflow at runtime. With the integration of ActiveMQ as the JMS messaging system, these parameters were even invalid due to broken links to LDAP objects that no longer existed. Therefore, the XML definition of the JMS connector will now be updated by the migration routine.

An automatic migration routine performs this task during the upgrade configuration.

You can run this routine by hand at any time.

1.4.4.3. Deletion of the Certificate Handler adaptor in the IdS-J

The Certificate Handler is automatically deleted during the upgrade configuration.

It was used in the pre V8.2C versions for the Windows Password Listener. In V8.2C, the feature to distribute certificates and the message server list was assumed by the Configuration Handler adaptor. Therefore, in V8.3, this adaptor is no longer necessary.

1.4.4.4. Issues Relevant for Upgrade from V8.3 or V8.3 R2

This section comprises all issues relevant for V8.3 or V8.3 R2 only.

1.4.4.5. JMS Audit Handler

The JMS-Audit handler is neither configured nor updated automatically as part of the normal installation and configuration. See the chapter in the Installation Guide to upgrade the JMS audit handler.

1.4.5. Issues Relevant for Upgrade from V8.4

This section comprises all issues relevant for V8.4 only.

1.4.5.1. Java-based Server

There are two new special adaptors responsible for the consumption of scheduler and Identity Manager messages:

- The **Entry Change Start Workflow Listener** and
- The **Provisioning Request Start Workflow Listener**.

The Entry Change Start Workflow Listener (or Provisioning Request Start Workflow Listener) is always co-located with the Entry Change Listener adaptor (or Provisioning Request Listener adaptor) and starts event-based processing (or provisioning) workflows.

These new adaptors replace the legacy adaptor **Start Realtime Workflow Listener**. There are no manual steps necessary.

1.4.6. Issues Relevant for Upgrade from V8.5

This section describes all issues relevant for V8.5 only.

1.4.6.1. Use of SSL

In V8.6, SSL handling has been extended. For ActiveMQ, only client-side SSL is now supported. As a result, you need to perform the manual steps described in the chapter “Manual Migration” to migrate Windows Password Listeners that connect with SSL to ActiveMQ to connect with client-side SSL.

1.4.7. Issues Relevant for Upgrade from V8.6

This section describes all issues relevant for V8.6 only.

1.4.7.1. Clear Text Passwords

In V8.7, the use of clear text passwords has been minimized. In older versions, the following features stored clear text passwords in LDAP:

- Passwords for JMS-based auditing
- Passwords for sending e-mail notifications in request workflows
- Passwords for sending notification e-mail in Tcl-based workflows

Now these passwords are stored encrypted. See the relevant sections in the chapter “Manual Migration” for details on how to migrate the passwords formerly stored as clear text passwords.

1.4.8. Issues Relevant for Upgrade from V8.7

This section describes all issues relevant for V8.7 only.

1.4.8.1. Topic Prefixes

In V8.9, target system-specific adaptors were introduced. In this context, the topic prefix name of the following listeners was changed:

Listener	Old Topic Prefix	New Topic Prefix
SetAccountPasswordListener	dxm.setPasswordRequest	dxm.setPasswordRequest._default
ProvisioningRequestListener	dxm.request.provisionToTS	dxm.request.provisionToTS._default
ProvisioningRequest StartWorkflowListener	dxm.request.workflow.provisionToTS	dxm.request.workflow.provisionToTS._default

If you upgrade from an older version, for example, V8.5, these changes were not made in all cases automatically. So, you must make the canges manually with Web Admin.

1.4.9. Issues Relevant for Upgrade from V8.9

This section describes all issues relevant for V8.9 only.

1.4.9.1. Client-side SSL Support in Realtime Workflows

In V8.10, client-side SSL for realtime workflows using either an ADS Connector or LDAP Connector or RACF Connector is supported. Additional connection parameters for this feature need to be put into the XML workflow definition. There are no manual steps necessary.

1.4.9.2. New LDAP Controls LDAP_CTRL_NO_MODTIME_UPD (1.3.12.2.1107.1.3.2.12.9) and LDAP Conditioned Operation Control (1.3.12.2.1107.1.3.2.12.10)

In V8.10, the LDAP lock mechanism has been revised and improved. For better performance, two new LDAP controls are implemented by the DirX Directory server V8.9 (9.4.454 or higher). These controls are set at the LDAP root in the LDAP attribute **supportedControl** (DAP attribute name: **SCON**) if you installed DirX Directory from scratch. If you just do a DirX Directory upgrade installation, these LDAP controls are not set at the LDAP root.

DirX Identity checks at configuration time whether the current DirX Directory version supports these controls. If the LDAP controls are supported, then it updates the LDAP-root automatically if missing. But keep in mind that manual migration is necessary if you decide to upgrade the DirX installation later.

1.4.9.3. Topic Prefixes

In V8.9, target system-specific adaptors were introduced. In this context, the topic prefix name of the following listener is automatically changed in V8.10:

Listener	Old Topic Prefix	New Topic Prefix
ImportToIdentity	dxm.request.importToIdentity	dxm.request.importToIdentity._default

2. Migration Procedure

The general migration procedure is:

1. Perform the **preparation steps** in the next chapter.
2. **Install DirX Identity** (upgrade installation). For details about installation see the chapter “Installing DirX Identity” in the *DirX Identity Installation Guide*.
3. **Configure DirX Identity** (upgrade installation). **Configure Connectivity Schema and data** (on the machine were the connectivity database resides). Afterwards **configure Provisioning Schema and data** (on the machine were the provisioning database resides). Next, perform Initial Configuration for each existing ActiveMQ Message Broker. Afterwards run the Initial Configuration for the “primary” **C server** (the first installed one in a distributed Identity environment). If necessary, the JMS subscription migration is done during this step. For details about installation, see the chapter “Installing DirX Identity” in the *DirX Identity Installation Guide*.

   Configure DirX Identity (this includes the automatic migration procedures). Use the Initial Configuration wizard for this step and perform the initial configuration for all existing domains. For details see the chapter “Configuring DirX Identity” in the DirX Identity Installation Guide.

   Every “Domain Configuration” migrates the Request Workflow/Activity instances.
4. Perform the manual migration steps in this manual.
5. Test all of your applications and workflows thoroughly to be sure that everything works well.
6. If you encounter any problems, read the "Known Problems" section in this manual.

3. Preparing the Migration

These steps are necessary to prepare the migration:

- Backup all DirX Identity databases to be able to reset to the starting point if something goes wrong.

Depending on the features you used, you need to perform the steps described in the next chapters.

3.1. Preserving Files

The installation procedure defines a set of paths which are backed up by the installer. For more information, see Configurable Content.

If you changed default parameters in files that are not backed up automatically, back them up yourselves. Examples of these changes are:

- idmsvc.ini - you increased, for example, memory requirements (IdS-J server)
- dxi.cfg - you made customizations according to the user documentation
- runserver.bat or runserver.sh - you increased, for example, memory requirements
- dxmmsssvr.ini - you used different port parameters, passwords or a **certdb** file path
- domain.xml - you set a different search timeout limit than the default
- bindcredentials.xml - you set your own timeout limits
- password.properties - you provided PIN values for encryption or signature (IdS-J server) or SSL keystore password.The file is written anew by the Configurator, which asks you to set your PIN values.
- gen*Keystores.bat/sh - scripts to generate keys; check for changed parameters
- set_Environment.bat/sh - scripts to generate certificates and keys; check for changed parameters
- web.xml - you changed parameters like SSL or debug
- cacerts - you set up certificates for SSL in this file in the Java JRE folder *dxi_java_home***/lib/security**.When creating the cacerts backup, ensure that your backup is outside *dxi_java_home*.

3.2. Customized Web Center

Create a backup copy of the related folders.

3.3. Provisioning Web Services and Related Clients

Create a backup copy of the file *install_path***/provisioningServices/spmlv2/conf.xml**.

Perform these steps when upgrading from a Version lower than V8.5:

- Create a backup copy of the related folder *install_path***/provisioningServlet**.
- Stop Tomcat (if you deployed said services into Tomcat) or Java-based server (otherwise), respectively.
- Remove the file **ProvisioningService.xml** from the folder *tomcat***/conf/Catalina/locahost** or from *install_path***\ids-j-***domain-***S***n/***tomcat/conf/Catalina/localhost**, respectively, in order to undeploy the obsolete servlet instance.

Perform this step when upgrading from Version V8.5 or higher:

- Create a backup copy of each deployment instance folder of the folder *install_path***/provisioningWebServices**, that is each sub-folder **provisioningServlet-***technical-domain* or **provisioningServlet-embedded-***technical-domain,* respectively,

3.4. Update Tomcat and Java Environment

This version runs with Java SE 11 only.It supports Tomcat version 9 only.

Perform these steps:

- Check your installed Java version.If version 11 is not yet installed, download this version with the latest security fixes and install it.
- Check your installed Tomcat version.If it is not a supported version, download a supported version with the latest security fixes and install it.Use the Java 11.
- Configure each Tomcat you designed for deployment of DirX Identity Web Applications so that it uses Java 11 at runtime.

3.5. Service Configuration Cleanup Linux

DirX Identity Java-based Worker Containers have been supported with DirX Identity Versions 8.2 but they are no longer supported with this version.

Please check whether these conditions are true:

- If your installation is an upgrade installation of these DirX Identity versions.
- You have already executed the integration utility **updrcs-linux.sh** in the folder *install_path***/etc** in order to integrate start/stop scripts into the operating system.An indication for this is the existence of the file **updrcs-linux.sh.log** in the same folder.

If these conditions are true, run the steps described in the section “Undoing the Integration” in the section “Integrating Start/Stop Scripts into the Linux Operating System” of the chapter “Configuring DirX Identity” of the *DirX Identity Installation Guide*.Here, the relevant document set is the one that corresponds to the DirX Identity version you intend to upgrade.

4. Automatic Migration during Configuration

Now you can start the initial configuration.

During configuration of DirX Identity, an automatic procedure runs and performs various steps that depend on the version you are migrating from. See chapter 1 for more information.

You can run some of the migration steps later on as separate tasks. See chapter 5 for more information.

5. Manual Migration

Perform all the steps described in this chapter that are relevant to adapting your environment to DirX Identity.

5.1. General Aspects

The following manual migration steps are relevant for all DirX Identity versions.

5.1.1. Restoring Preserved Files

Restore the specific information from the files you backed up as described in the section “Preserving Files” in the chapter “Preparing the Migration” by merging the information in the backup files into the files in the relevant locations.

You only need to update the file *dxi_java_home*/**lib/security/cacerts** with the information from the relevant backup.

5.1.2. Migrating Passwords for JMS-based Auditing



Performing this task is only necessary if you use JMS-based auditing; that is, if the **JMS-based Auditing** flag is checked in the Status and Auditing tab of your IdS-J Server (Java-based Server) configuration object in the Connectivity database. (Connectivity view → DirX Identity Servers → *your-ids-j-server* → Status and Auditing tab → Auditing – General section).

In older DirX Identity versions, the password and user were taken from an attribute at the ids-j object where the password was stored in clear text. Beginning with V8.7, the user and password are now taken from a specified bind profile. If you use JMS-based auditing, you need to migrate the configured user and password as follows:

1. Create a bind profile that holds the user and password. We recommend adding the bind profile to your Identity Store connected directory. Only the **User** and **Password** bind parameters are required. Enter the text **JMS** in **Anchor**.
2. In the IdS-J Server’s Status and Auditing tab, enter the newly created bind profile into **Bind Profile**.
3. Restart the Ids-J Server.
4. Repeat steps 2 and 3 for every IdS-J Server configuration object that uses JMS-based auditing using the bind profile you created in step 1.

5.1.3. Migrating Passwords for Mail and SMS Gateway Configuration

In the Provisioning view, the **SMTP** configuration object for sending e-mail notifications and the **SMS Gateway** configuration object for sending SMS notifications are located in **Workflows** → **Configuration** → **Services**.

If the **Authenticate** flag is checked for these configuration objects, the password

information for these objects needs to be migrated.

For each configuration object, enter the password in the relevant field and then save the object. Now the passwords are stored encrypted according to the type of encryption mode specified in the Connectivity database's central configuration object (in **Encryption Mode** in the Server tab).

5.1.4. Migrating Passwords for Sending Notifications in Tcl-based Workflows

You can define notifications in the Connectivity view at the **metacp** job in the Operation tab. If you have used these notifications with a user and password, you need to migrate the referenced service object, which is usually the Mail Service (**Configuration** → **Services** → **System** → **Mail Service**).

Open the notification object (it's usually located below the **metacp** job object (Connectivity view → **Jobs** → *your_metacp_job* → *your_notification_object*)) and then click the **Service** link in the Service Definition area to open the service object. Enter the password and save the object. Now the password is stored encrypted according to the encryption mode defined in the Connectivity central configuration object (**Encryption Mode** in the Servers tab).

5.1.5. Using SAP JCo version 3.1.4 or Higher for SAP ECC UM Connector/Agent

We recommend upgrading to the latest version.

5.1.6. Migrating SPML Provisioning Web Services

After installing and configuring DirX Identity, configure the services according to the section "Runtime Operation" in the chapter "SPML Provisioning Web Services" in the *DirX Identity Integration Framework Guide*.

After customizing the Provisioning Web Services, merge the relevant configuration information from your backup into the relevant files of the folder *install_path*/*provisioningWebServices/provisioningServlet-**technical_domain_name*.

For selected migration paths, the files **ProvisioningWebServicesChanges**.txt* in the subfolder **Documentation/DirXIdentity/ProvisioningWebServices** of the installation media list all files and folders to be merged.

When upgrading from a version lower than V8.5, the folder *install_path*/**provisioningServlet** is obsolete. Keep a related backup outside the DirX Identity installation and then remove this folder. When you are sure that the backup is no longer needed, remove the backup, too.

5.1.7. Migrating SPML Provisioning Web Services Clients

Migrating the SPML Provisioning Web Services clients consists of the following steps:

- Stop using the obsolete, unsupported sample client and remove the following related

files: **agentconf.xml**, **conf.xml**, **cpappend.bat**, **fireSOAP-Request.bat**, **fireSOAP-Request.sh**, **log4j.properties**, **request.xml** in the folder *install_path/***provisioningServices/lib**. These files represent an obsolete sample client that uses a proprietary format in sent requests and received responses. Using this client is no longer supported. Stop using this client or related customizations and then remove these files.

- Update the runtime environment of your Web Services clients with the jar files shipped with DirX Identity. The set of jar files to be used is located in the folder *install_path/***provisioningServices/lib**.
- Ensure that the classpath in your runtime environment of your Web Services clients is computed so that the appropriate set of jar files is on the classpath. You'll find samples of the correct classpath computation in the file *install_path/***provisioningServices/spmlv2/fireSpmlv2-Request.bat** (Windows platforms) or **fireSpmlv2-Request.sh** (UNIX platforms).
- The file *install_path***/provisioningServices/spmlv2/conf.xml** is overwritten at installation time. Therefore, you must adapt this file according to the section "Getting Started with the Test Client" in the chapter "SPML Provisioning Web Services" in the *DirX Identity Integration Framework Guide*. You can do this either from scratch or by merging the relevant information from a related backup.

5.1.8. Migrating a Customized Web Center

If you have a customized Web Center, you must merge the changes from your backed-up customized version into the new version of Web Center which now contains the domain name:

*install_path***/web/webCenter-***technical_domainname*

The files **WebCenterChanges*.txt** in the subfolder **Documentation/DirXIdentity/WebCenter** of the installation media list all files and folders to be merged.

Note that the new URL is now

http://host**:8080/webCenter-***technical_domainname*

You can rename the URL by renaming the related file **webCenter-***technical_domainname***.xml** in the folder *tomcat_path***/conf/Catalina/localhost**. If you rename the URL, you must update the related URL in the Provisioning Store accordingly. (DirX Identity Manager → Provisioning → Workflows → Configuration → Services → Approval, Property **Location (URL, server)**).

5.1.9. Migrating a Customized Web Center for SAP NetWeaver

If you have a customized Web Center for SAP NetWeaver, you must merge the changes from your backed-up customized version into the new version of Web Center which now contains the domain name:

*install_path***/web/webManagerForSAP-***technical_domainname*

The files **WebCenterChanges*.txt** in the subfolder **Documentation/DirXIdentity/WebCenter** of the installation media list all files and folders to be merged.

5.1.10. Migrating a Customized Web Center for Password Management

If you have a customized Web Center for Password Management, you must merge the changes from your backed-up customized version into the new version of Web Center which now contains the domain name:

*install_path***/web/pwdManagement-***technical_domainname*

The files **WebCenterChanges**.txt* in the subfolder **Documentation/DirXIdentity/WebCenter** of the installation media list all files and folders to be merged.

5.1.11. Migrating a Customized Business User Interface

See the *Business User Interface Configuration Guide* for more detail information about migrating.

5.1.12. Migrating the ActiveMQ Messaging Server

In some cases, the migration of the repository (file-based database kahadb) from a former ActiveMQ version to the version that comes with DirX Identity V8.10 does not work.

For this reason, we strongly recommend that you verify that all message queues in ActiveMQ are empty before upgrading (enqueuer and dequeuer counters are equal in Web Console). In rare cases, ActiveMQ doesn't start correctly after migration because of kahadb issues (the repository). In this case, the only choice is to delete the kahadb completely.

5.1.13. Adapting Size Limits for LDAP Searches (using DirX Directory)

The size limit applied to an LDAP search operation limits the number of returned entries in single searches and paged searches.

The source is:

1. LDAP search operations option size limit (user defined)
2. LDAP server's configuration property "**Ldap-Search-Svc-Ctl**" (**LSES**) (defined by the administrator for each LDAP server)
3. DSA's User Policy (defined by the administrator, user specific)

The value in force is 3 or a smaller limit from 2 or 1 (if there is no specific limit for that user; for example, there is no DSA User Policy for that user).

Older directory versions (before release "DirX Directory V8.2B") had a problem interpreting the size limit in search operations: they applied the size limit to every single search operation in a series of paged searches. Therefore, the size limit could easily be circumvented by specifying a page size smaller than the size limit in force.

Starting with DirX Directory 8.2B, the size limit is applied to the entire search operation regardless of whether or not it uses simple paging control.

Caution: Applications may see a changed behavior:

- Retrieving the n^{th} page of a paged search operation may return with a “size limit reached” return code and an empty cookie.
- The size limit in force must be big enough for all entries in all pages returned in one search operation.
- For anonymous users, the size limit is defaulted to 2048.
- Use the UserPolicy (USP attribute of the root DSE) to set the required limits.

Please note that DirX Identity has defined user policies with a size limit of 32,768 for non-paged searches and a paged result size limit of 100,000 (for the Connectivity branch and for each domain in the Provisioning branch).

Example (including the sample domain):

```
dirxadm> show / -attr USP -p
1) /
     User-Policy
         User-Subtree-Name       :
/DXMC=DirXmetahub/DXMC=Groups/CN=Read
         Size-Limit              : 32768
         Paged-Search-Size-Limit : 100000
     User-Policy
         User-Subtree-Name       : /CN=DirXmetaRole-
SystemDomain/CN=SystemAdmin
         Size-Limit              : 32768
         Paged-Search-Size-Limit 100000
     User-Policy
         User-Subtree-Name       : /CN=My-
Company/CN=TargetSystems/CN=DirXmetaRole/CN=Groups/CN=DomainAdmins
         Size-Limit              : 32768
         Paged-Search-Size-Limit : 100000
```

Normally you should not have any size limit problems; if you do, you must increase either the size limit of 32,768 or the paged result size limit of 100,000 to some higher value.

When configuring a new customer domain, the default values for the two size limits are as described in this section.

5.1.14. Extending the DirX Identity Schema for Customer Domains

If you upgrade to a new DirX Identity version, be aware that the schema for certain target system objects may have been extended for the purpose of provisioning new connected system attributes. The schema for the My-Company domain is updated during the Initial Configuration if the domain step with the My-Company domain is selected. If you configure your own customer domain, you must update the schema for that domain by running the appropriate agent schema scripts provided in the *install_path*/schema/tools* folder. For a detailed description of this procedure, see the section “Extending the Schema for the Target System Workflows” in the *DirX Identity Application Development Guide*.

The following sections provide an overview of the schema changes provided for all the supported connected systems. Except for RACF, no migration needs to be done manually.

5.1.14.1. Agent Schema Changes in V8.10

There are no agent schema changes in V8.10.

5.1.14.2. Agent Schema Changes in V8.9

The following systems are no longer supported:

- UNIX-PAM
- SoarianClinical

5.1.14.3. Agent Schema Changes in V8.7

There are no agent schema changes in V8.7.

5.1.14.4. Agent Schema Changes in V8.6

5.1.14.4.1. RACF

The following attributes had an erroneous substring matching rule (caseExactSubstringMatching instead of caseIgnoreSubstringMatching)

- krbPrincipalName
- racfLdapBindDN
- racfLdapBindPw
- racfNdsUserName
- racfLNotesShortName

An automatic migration of these attributes is available and is executed while running the DirX Identity Configuration tool. You can also start this migration step by hand at any time. It is located in:

install_path/**tools/migration/86/database**

As the Provisioning part is migrated, all arguments refer to the Provisioning configuration

database.

Usage:

MigrateRACFschema.bat *host port user password ssl domain-DN logfile*

Example:

```
MigrateRACFschema.bat localhost 389 "cn=DomainAdmin,cn=My-Company"
dirx 0 "cn=My-Company" log.txt
```

Manual schema update:

After the user data has been migrated, you must drop the erroneous attributes manually from the schema. The erroneous attributes have been renamed in the schema by appending the suffix **OLD**. Adapt the relevant object classes as listed in the following files:

install_path/**tools/migration/86/database/postSchemaUpdate/dirx.racf.ldif** (for DirX Directory)

For DirX Directory, the relevant changes are:

```
# "racfLNotesShortNameOLD" has been dropped
dn: cn=schema
changetype: modify
add: objectclasses
objectclasses: ( 1.3.12.2.1107.1.3.102.6.16.16 NAME
'racfLNotesSegment'
    DESC 'This is a proprietary DirXmetahub objectclass.'
    AUXILIARY
    MAY ( racfLNotesShortName ) )
-
# "racfNdsUserNameOLD" has been dropped
dn: cn=schema
changetype: modify
add: objectclasses
objectclasses: ( 1.3.12.2.1107.1.3.102.6.16.17 NAME 'racfNdsSegment'
    DESC 'This is a proprietary DirXmetahub objectclass.'
    AUXILIARY
    MAY ( racfNdsUserName ) )
-
# "racfLdapBindDNOLD" has been dropped
# "racfLdapBindPwOLD" has been dropped
dn: cn=schema
```

```
changetype: modify
add: objectclasses
objectclasses: ( 1.3.12.2.1107.1.3.102.6.16.18 NAME
'racfProxySegment'
    DESC 'This is a proprietary DirXmetahub objectclass.'
    AUXILIARY
    MAY ( racfLdapBindDN $ racfLdapBindPw $ racfLdapHost ) )
-
# "krbPrincipalNameOLD" has been dropped
dn: cn=schema
changetype: modify
add: objectclasses
objectclasses: ( 1.3.12.2.1107.1.3.102.6.16.19 NAME
'racfKerberosInfo'
    DESC 'This is a proprietary DirXmetahub objectclass.'
    AUXILIARY
    MAY ( racfCurkeyVersion $ racfEncryptType $ krbPrincipalName $
    krbPrincipalNameOLD $ maxticketage ) )
-
```

5.1.14.4.2. Salesforce

The following new attributes are provided:

- dxmSLSFcontactId
- dxmSLSFcontactName
- dxmSLSFdefaultPermissionSet
- dxmSLSFdefaultPermissionSet-DN
- dxmSLSFpermissionSet
- dxmSLSFpermissionSet-DN
- dxmSLSFprofileId
- dxmSLSFprofileName
- dxmSLSFprofile-DN
- dxmSLSFuserLicenseName

The following new object class is provided:

dxmSLSFpermissionSet

Please note that currently only the schema for Salesforce has been extended. These attributes and object classes are currently not supported by the Salesforce Connector and the Salesforce realtime workflow.

5.1.14.5. Agent Schema Changes in V8.5

5.1.14.5.1. ADS

The following new attributes are provided:

- dxmLyncInternetAccessEnabled
- dxmLyncOptionFlags
- dxmLyncPrimaryHomeServer
- dxmLyncUserPolicies
- dxmLyncUserRoutingGroupId
- dxmLyncUserEnabled
- dxmLyncFederationEnabled
- dxmLyncDeploymentLocator
- dxmLyncPrimaryUserAddress

The following new object class is provided:

dxmLyncUser

5.1.14.6. Agent Schema Changes in V8.4

DSWin is no longer supported.

NDS is no longer supported.

Salesforce is new.

5.1.15. LDAP Lock Mechanism for User Objects

A lock mechanism (preventing parallel updates for a user) has been implemented in DirX Identity V8.6 and was revised in DirX Identity V8.6-SP1 and again in DirX Identity V8.10. More details are available in the use case document *DXI Java Programming*.

5.1.16. Migrating Realtime Workflows to Support Client-side SSL Authentication

Client-side SSL authentication is supported in realtime workflows when using one of the following connectors: ADSConnector, LDAPConnector, and RACFConnector.

The XML workflow definition needs to be adapted to support additional connection parameters.

Migration is done automatically during the first Configurator run. All workflows in the Connectivity database (using any of the connectors listed above) are processed.

You can start this migration step by hand at any time. It is located in:

install_path/**tools/migration/89/rtworkflows**

As the Connectivity part is migrated, all arguments refer to the Connectivity configuration database.

Usage:

MigratePortForClientSSL.bat *host port user password ssl logfile*

or

MigratePortForClientSSL.sh *host port user password ssl logfile*

Example:

MigratePortForClientSSL.bat localhost 389 "cn=admin,dxmC=DirXmetahub" dirx 0 trace.txt

5.2. Aspects Relevant to Using DirX Directory V8.6 or Higher

5.2.1. Rebuilding Attribute Indexes to Handle Improved Search Operations

DirX Directory V8.6 provides improved search operation processing, especially for AND filter combinations that begin with filter items (for example, **uid=b***), “greater than or equal to” filter items (for example, **time>=20170716**) and/or “less than or equal to” filter items (for example, **time⇐20170716**) - apply.

For best performance, we recommend re-building the indexes for the attributes where these search filters apply on a regular basis; for example, on a bimonthly schedule.To perform this task, for example, use the following **dirxadm db attrconfig** command:

dirxadm> db attrconfig uid time -index BUILD

Note that the DSA runs in the POSTINDEXING operation mode while the db attrconfig command is in progress and update operations will be rejected.

DirX Identity attributes that need to be handled this way include:

dxmStatusExpirationTime

dxrCertificationDate

dxrDeleteDate

dxrDisableEndDate

dxrDisableStartDate

dxrEndDate

dxrExpirationDate

dxrPwdExpiryNotified

dxrStartDate

5.2.2. Migration of New LDAP Controls LDAP_CTRL_NO_MODTIME_UPD (1.3.12.2.1107.1.3.2.12.9) and LDAP Conditioned Operation Control (1.3.12.2.1107.1.3.2.12.10)

Starting with DirX Identity V8.10, the Entry Lock Manager has been improved and makes use of two new LDAP controls. These LDAP controls are available with DirX Directory Server V8.9 (9.4.454 or higher) if you installed DirX from scratch. A DirX upgrade will not set the LDAP controls at the LDAP root.

The DirX Identity Configurator sets these LDAP controls.But please note that a manual migration is necessary if you decide to upgrade the DirX installation after you have already configured DirX Identity.

In this scenario, the migration procedure is as follows:

- Set the password of the DirX administrator (Tcl variable DIR_PW) in the file *install_path*/basic.input.tcl*.
- Open an MS-DOS command prompt or a UNIX shell in the directory *install_path*/schema/role-dirxee/SystemDomain*.
- Run the following command: **dirxadm init.v810.adm**.
- Check the content of the tracefile **setup-trace.txt** in the current directory.The tracefile should show that **dirxadm** terminated with exit code 0.
- Set the TCL variable DIR_PW in **basic.input.tcl** to “”.

5.3. Aspects Relevant to DirX Identity V8.10

5.3.1. Reports based on TIBCO Jaspersoft

Starting with V8.10, images, styles, and nationalized texts (i18n) for Jaspersoft-based reports are stored in LDAP.Therefore, the configuration for the reports must be extended.This can be done manually, report per report, or automatically with a migration script.

For details, see the chapter "Upgrading" in the use case document *DXI Jaspersoft Reports*.

5.3.2. Support of Java 11 and Tomcat 9

This version runs with Java 11 (JRE or JDK) and Apache Tomcat 9 only.

5.4. Aspects Relevant to DirX Identity V8.9

5.4.1. Support of Java SE 11

This version runs with Java SE 11 only.Note that the Identity installation no longer offers the use of an embedded Java JRE runtime.The Java environment must be externally provided.

For customer-specific code that is written in Java, the customer should verify if the code is running with Java SE 11 without rebuilding it.Note that Java SE 11 no longer contains some web service APIs that were included in Java JRE 8. A rebuild with Java 11 and some additional third-party jar files may therefore be necessary.

See the Oracle's documentation of removed APIs, features and options under https://www.oracle.com/technetwork/java/javase/9-removed-features-3745614.html and https://www.oracle.com/technetwork/java/javase/10-relnote-issues-4108729.html#Removed.

See also the section about Class Loading in IdS-J server in this guide.

5.4.2. Support of Apache Tomcat 9

This version supports Apache Tomcat 9 only. Additionally, for the DirX Identity Web services that are deployed in Tomcat, the Tomcat server must run with Java SE 11.

In typical datacenter scenarios, DirX Identity's Web Center or Business User Interface is used behind a reverse proxy; for example, a firewall or load balancer. To configure the Tomcat container as a reverse proxy target, see the Tomcat documentation under https://tomcat.apache.org/tomcat-9.0-doc/proxy-howto.html.

5.4.3. JMX Access

In this version, the JMX access of the IdS-J server and the Apache ActiveMQ server have been pre-configured so that user credentials are always needed.

See the relevant sections "JMX Access to the Java-based Server" and "JMX Access to the Message Broker" in the *DirX Identity Connectivity Administration Guide*.

5.4.4. Class Loading in IdS-J Server (for Customer-specific Request Workflow Job Implementations)

The Java class loading in the IdS-J server has been simplified. In previous releases, when deploying a customer-specific request workflow job implementation (in the folder "*install_path*/**ids-j-**...**/confdb/jobs/**...") you had to supply the jar file of your job implementation and several other jar files, such as dxrOrder.jar, dxrPolicies.jar, dxrServices.jar, and so on.

This approach could cause runtime problems (class loading problems) as these additional jar files existed at different locations in the installation, too. Very often there were conflicts with the Jar files loaded from *install_path**/ids-**...**/extensions/com.siemens.idm.domcfg/lib*).

Therefore the whole environment was simplified:

- Jar files such as dxrOrder.jar, dxrPolicies.jar, dxrServices.jar etc. exist only once in

*install_path**/ids-**...**/confdb/common/lib*.

- Jar files such as dxrOrder.jar, dxrPolicies.jar, dxrServices.jar have been dropped from *install_path***/ids-j-**...***/confdb/jobs/**... and *install_path**/ids-j-**...**/extensions/com.siemens.idm.domcfg/lib*.
- Some jar files have been copied from *install_path**/ids-**...**/lib* to *install_path**/ids-**...**/confdb/common/lib*.

Now the request workflow job implementation folders (*install_path***/ids-j-**...***/confdb/jobs/**...) must only contain the jar file of the job implementation (and optionally some other jar files that only that job requires; for example, third-party jar files).

The job folders provided with the installation now look as follows:

- *install_path***/ids-j-**...**/confdb/jobs/builtin/lib**
  - builtinImpl.jar
- *install_path***/ids-j-**...**/confdb/jobs/CertificationCampaign/lib**
  - dxrCampaignController.jar
- *install_path***/ids-j-**...**/confdb/jobs/consistencyCheck/lib**
  - dxrConsistencyCheck.jar
- *install_path***/ids-j-**...**/confdb/jobs/eventBasedRules/lib**
  - dxrEventBasedRules.jar
- *install_path***/ids-j-**...**/confdb/jobs/framework/lib**
  - various implementations of framework jobs (used by realtime workflows)
- *install_path***/ids-j-**...**/confdb/jobs/LdifChange/lib**
  - LdifChangeImpl.jar
- *install_path***/ids-j-**...**/confdb/jobs/metahubworkflow/lib**
  - metahubworkflowImpl.jar
- *install_path***/ids-j-**...**/confdb/jobs/order/lib**
  - orderImpl.jar
- *install_path***/ids-j-**...**/confdb/jobs/RiskGovernance/lib**
  - dxrRiskGvnController.jar
- *install_path***/ids-j-**...**/confdb/jobs/setpassword/lib**
  - setpasswordImpl.jar
- *install_path***/ids-j-**...**/confdb/jobs/siemensgid/lib**
  - siemensgidImpl.jar
- *install_path***/ids-j-**...**/confdb/jobs/ticketControl/lib**
  - dxrTicketControl.jar

Your tasks are to:

- Provide the lightweight job folders as shown above for your customer-specific jobs and drop all the jar files that are already available in *install_path*/ids-**...**/confdb/common/lib*.
- Move your customer-specific implementations of user hooks or any other interfaces to *install_path*/ids-**...**/confdb/common/lib*.

5.4.5. Handling of Escaping in Provisioning Rule Filters (RJYNS3)

In former versions, sometimes you ran into trouble if you wanted to create a filter with special characters in the condition; for example, description=a(b)c,. Therefore, the internal handling of this filter was changed. Provisioning rule filters are now stored escaped in LDAP except for the time constructs. Asterisks * in conditions are stored as **. This means they are used as wildcards. If you want to check for an asterisk, use *\2a** (insert **\2a** in the LDAP tab of the filter editor).

Automatic migration does not take place because it's not always clear what the filter should represent. DirX Identity provides a standalone tool that searches for rules that might cause trouble in the new version. Run the tool and then check the output and look at the rules that are reported. Change the rule using DirX Identity Manager if necessary. But in every case, do a save for this rule to be sure the rule is stored correctly. The example given here is stored in the following way: (description=a\28b\29c).

On Windows, call:

*install_path***\GUI\tools\provrulechecker\CheckProvisioningRules.bat**

First specify the bind credentials and the root for the Provisioning rules to be checked in **CheckProvisioningRulesConfig.xml**.

For details, see **README.txt** in the aforementioned directory.

5.4.6. Old and New Delegations

As of version 8.9, DirX Identity supports a new type of delegations which are easier to understand and to use than the old delegations. You can either continue to use the old delegations or switch to the new ones, but you cannot use them both in parallel. This section gives an overview of the differences between the old and new delegations and describes what to pay attention to when switching from the old to the new delegations (or vice versa).

5.4.6.1. Old Delegations

The old delegations support all operation types (create, grant, approve, delete, etc.) They can be restricted to specific subsets of resources, like grant all roles, or approve group A and group B.

You can view the delegations in the DirX Identity Manager:

- Select view Policies.
- Open folder Delegations.

The old delegations have no operation (dxrOperationImp) but have one or more access right links.

The old delegations can be managed in DirX Identity Web Center. They are not supported by the DirX Identity REST Services and cannot be managed in the DirX Identity Business User Interface.

5.4.6.2. New Delegations

The new delegations support only operation types grant and approve. They cannot be restricted to specific subsets of resources.

You can view the delegations in the DirX Identity Manager:

- Select view Policies.
- Open folder Delegations.

The new delegations have operations (dxrOperationImp) grant or approve but no access right links.

The new delegations are supported by the DirX Identity REST Services and can be managed in the DirX Identity Business User Interface but not in DirX Identity Web Center.

5.4.6.3. Switching Between Old and New Delegations

A global flag defines which type of delegation is enabled. You can view and set the flag in DirX Identity Manager:

- Select view Domain Configuration.
- Open the domain object.
- Select tab Policies.

If the flag **Delegation Assignment stores Operation** is checked, the new delegations are enabled. If it is not checked, the old delegations are enabled.

After installation and configuration of V8.9, the flag is not checked, so the old delegations are still enabled.

When activating the new delegations, the old delegation assignments and access right entries are not deleted but they will be ignored. The Business User Interface will not display them. The Identity REST Services will also ignore them.

Switching back from the new to the old delegations is not recommended. If you do it anyway, note that the delegators will still see their (now deactivated) new delegations in Web Center but Web Center will not handle the delegations correctly. Therefore, the delegators will have to delete them in Web Center, or you delete all new delegations via the DirX Identity Manager.

5.4.6.4. Access Policy Evaluation

Access policy evaluation always takes delegations into account according to the global flag. If the new delegations are enabled, the old ones are ignored. If the old delegations are enabled, the new ones are ignored.

5.4.6.5. Migrating to DirX Identity 8.9

- If you don't want to use delegations at all:
- Nothing to do.
- If you want to use the old delegations:
- Nothing to do.
- If you want to use the new delegations:
- Check the global flag.
- If you've used the old delegations before:
- Tell your users that any previous delegation is no longer valid.
- Delete the old delegations or tell your users to delete them.
- Tell your users how to manage new delegations via the Business User Interface.

5.5. Aspects Relevant for Upgrade from V8.6

This section describes all aspects relevant to upgrading to the current version from DirX Identity V8.6.

5.5.1. Replacement of REST Approval Service with REST Service

The REST Approval Service has been replaced with the new DirX Identity REST service which includes more functionality.Web applications must be adapted.You'll find the new service in the installation under *install_path**/restServices.*

5.5.2. Replacement of HTML5-based Approval App with Business User Interface App

The HTML5-based Approval App has been replaced with the new DirX Identity Business User Interface which includes more functionality.As before, the new interface is configured with the Configuration Wizard.

5.5.3. Migrating Identity Server SSL Configurations

In V8.6, the setup and handling of SSL connections have changed slightly and includes the distribution of key material. This section describes the necessary steps to migrate existing SSL key material when upgrading from V8.3 and newer. Note that if you migrate from a version older than V8.3 and want to reuse existing key material, you must upgrade to V8.3 first and then perform the manual steps described in the section "Migrating Identity SSL Configurations" in the section "Aspects Relevant for Upgrade from V8.2".

Note that we recommend setting up new key material as it is described in the chapter “Securing Identity Server Connections with SSL” in the *DirX Identity Connectivity Administration Guide*.

5.5.3.1. Preparing Existing Key Material

The existing key material storage must be renamed. Go to the folder *install_path*/ssl* and rename following files:

- **server-keystore** to **identity-keystore** (private key of the server)
- **client-truststore** to **identity-truststore** (certificate of the server and CA)

Make sure that the CA certificate stored as *install_path*/ssl/ca.crt* is imported to *jre_root/*jre/lib/security/cacerts*. Note that *jre_root* is the path to the JRE used by DirX Identity components.

To import the **ca.crt** file, use the DirX Identity Manager. In the **Tools** menu, choose **Options**. On the next page, the Manager’s truststore is selected by default (“This application’s installation folder” is already selected and the file *install_path*/**GUI/cacerts** is shown). Choose **Java Runtime Environment**. Choose **Import** and then select the server certificate you want to import. When prompted for the keystore password, enter the default value **changeit**. Click **OK** and the certificate will be imported.

5.5.3.2. Generating New Client Key Material

DirX Identity V8.6 now requires the use of SSL client authentication for all ActiveMQ Message Broker clients when securing server connections with SSL globally. This means you need to provide key material for the SSL client authentication.

First, adapt the configuration for the generation scripts. Follow the instructions in the chapter “Configuring the Certificate Generation Scripts” in the *DirX Identity Connectivity Administration Guide*. The parameter values for **dname**, **validity** and **pwd** must be configured in the same way as they were before the upgrade.

Next, execute the generation scripts for the client key creation. Follow the instructions in the following sections of the *Connectivity Administration Guide* in this order:

- Setting up the Client Key
- Signing the Client Certificate Request
- Importing the Client Certificate into the Shared Key Store
- Converting the Client Key to PEM

As a result, you must now have at least the following files in the *install_path***/ssl** folder:

- **ca.crt**
- **ca-crt.pem**
- **client.crt**
- **client-key.pem**

- **server.crt**
- **server-key.pem**
- **identity-keystore**
- **identity-truststore**
- **password.properties**

5.5.3.3. Distributing Shared Key Material

The *install_path* must be available as the environment variable **DIRXIDENTITY_INST_PATH** when securing server connections with SSL globally. Make sure that the variable **DIRXIDENTITY_INST_PATH** exists. It can be missing on some hosts in distributed environments with a DirX Identity client component deployed into a native Tomcat. If the variable does not exist, define the environment variable **DIRXIDENTITY_INST_PATH** using an existing folder as a value.

If you have a distributed environment, you must copy and adapt (if necessary) the files from the source *install_path***/ssl** folder into the *install_path***/ssl** folder on the target machine.

If you installed a server (Java-based Server, C++-based Server, or ActiveMQ Message Broker), adapt the parameter value for *dname* in the configuration script *install_path* /**ssl/set_Environment.bat** and then perform the steps described in the following chapters of the *DirX Identity Connectivity Administration Guide* in this order:

- Setting up the Host Server Key
- Signing the Server Certificate Request
- Importing the Certificate into the Server Key Store
- Converting the Server Key Store to PEM

If you installed a client component (Windows Password Listener, Web Center, Identity Manager, and so on), copy the files

- **identity-keystore**
- **identity-truststore**
- **ca-crt.pem**
- **client-key.pem**
- **password.properties**

from a machine where a server is located into the local *install_path*/**ssl** folder.

5.5.4. Migrating Windows Password Listeners Connecting with SSL to ActiveMQ to Connect with Client-side SSL

With DirX Identity V8.6, ActiveMQ clients like Java-based and C++-based Servers, Web clients or Password Listener use client-side SSL when connecting to the ActiveMQ Message Broker if system-wide SSL is selected in the Configuration Wizard. Before this configuration is selected, the certificate-generating scripts must be executed as described in the chapter

“Setting up the X.509 Certificates” in the *Connectivity Administration Guide*.

If you upgrade from a DirX Identity V8.3, V8.4 or V8.5 version and you have Windows Password Listeners running using SSL to the Message Broker, you can migrate them one after another to use client-side SSL while the ones that haven’t been migrated yet are still running and connecting with server-side SSL by performing the following actions in the given order:

- Set to inactive the ConfigurationHandler adaptors of all Java-based Servers configured for domains related to those Password Listeners which are not yet to be upgraded. You must perform this task in the Data View because, with “Manage IdS-J Configuration”, you must leave one ConfigurationHandler adaptor for a given domain in the active state because they are selectable by radio buttons. You must deactivate these Configuration Handlers; otherwise, they will send the current Message Broker configuration to the Password Listeners as soon as the broker service is restarted after upgrading, but the old Password Listeners must talk to the “old” Message Broker configuration identified by the old port.
- Copy the **activemq.xml** file under *install_path*/**messagebroker/conf** to a separate location of your file system in order to save the old “transportConnector” section.
- Upgrade DirX Identity - including the Message Broker and the Java-based Server - to V8.6 and configure the system with SSL (because this is the use case for which a migration is needed). In the Message Broker step of the Configuration Wizard, specify a different port from the one the old Password Listeners are using.
- Stop the Message Broker service and then change the **activemq.xml** file by copying the saved “transportConnector” section under the “transportConnectors” section alongside the new “transportConnector”.
- Re-start the Message Broker. With the two transport connectors specified, it now listens on both ports: the old one configured with SSL and the new one configured with client-side SSL.
- Now you can upgrade each Password Listener step by step and configure it with the new port and with SSL as described in the chapter “Securing the Windows Password Listener with SSL” of the *Connectivity Administration Guide*. The V8.7 Windows Password Listener always uses client-side SSL if SSL is specified by “UseSSL=1” in the **options.ini** or **msgServers.ini** file even if the property “UseClientAuth” is set to 0.



you only need to perform these migration steps if your Password Listeners were configured with SSL. If not, you don’t have to do any migration regarding Password Listeners. You can just upgrade each one - after upgrading DirX Identity - while others with older versions (at least V8.3) can run in parallel.

5.5.5. Migrating Passwords for Web Admin and Active MQ Web Console

In V8.6, the passwords for the pre-defined **admin** account are stored hashed in files. If you then upgrade the relevant files, *install_path*/**ids-j**-*domain*-**S***n*/**tomcat/conf/tomcat-users.xml** and *install_path*/**messagebroker/conf/jetty-realm.properties** are not overwritten. So you need to hash the passwords yourself.

Web Admin Password

Go to *install_path*/**ids-j**-*domain*-**S***n*/**bin** and open a Windows command shell or a UNIX shell:

Usage:

dxidigest.bat/sh *password*

Example:

```
dxidigest.bat mypassword
Digesting a password using the specified algorithm (default: sha-512)
mypassword:ebe81bc72d9b04b143b6795882d12b3848f930bd4a39e2f4726ae1efb4
a6b971$1$373e057af2ea30e4ed000282b6af79ef2ba64e247c46c3c155fa40949952
f41947de75d2f38c3a377cac16c7b5d31ed6aec64d745698de5788417e575285959d
```

Copy the part after the colon and then insert it into the **tomcat-users.xml** file as the value for **password**.

```
<tomcat-users>
  <user name="admin" password="xxx" roles="admin,manager" />…
```

ActiveMQ Web Console Password

Go to *install_path*/**messagebroker**/**bin** and open a Windows command shell or a UNIX shell:

Usage:

```
dximqdigest.bat mypassword
2016-09-22 16:02:42.780:INFO::main: Logging initialized @114ms
mypassword
OBF:1uh41zly1x8g1vu11ym71ym71vv91x8e1zlk1ugm
MD5:34819d7beeabb9260a5c854bc85b3e44
Press any key to continue . . .
```

Copy the line with MD5 and insert it into the **jetty-realm.properties** file as

```
aadmin: MD5:34819d7beeabb9260a5c854bc85b3e44, admin
```

5.6. Aspects Relevant for Upgrade from V8.5

None.

5.7. Aspects Relevant for Upgrade from V8.4

None.

5.8. Aspects Relevant for Upgrade from V8.3

This section describes all aspects relevant to upgrading to the current version from DirX Identity V8.3.

5.8.1. Migrating External Messaging Client Tools

Existing instances of the External Workflow Starter and the Eventing tool from releases older than than this release are obsolete.Instances of these tools from service packs or hot fixes related to releases older than this one are obsolete, too.To continue using these tools, use the tools from this release by extracting and configuring the tools shipped with this release by hand.

5.8.2. Updating the JMS Audit Handler Deployment

If you have enabled the sending of audit records of DirX Identity Java-based Servers via JMS, you need to replace the deployment folder of the JMS Audit Handler with the new jar file(s).

The handler must be deployed into each IdS-J server into the following folder: *install_path* /**ids-j**-*domain*-**S*n***/extensions/com.siemens.idm.audit.jms*.

Replace the sub-folder **lib** with the one that is provided in the folder **Additions/jmsAuditHandler/com.siemens.idm.audit.jms.zip/lib** of the installation medium. The only required jar file is: **com.siemens.idm.audit.jms.JmsAuditLogHandler.jar**.

5.8.3. Displaying Inherited Privileges

Privileges can be assigned (direct, by rule or BO inheritance), inherited from other privileges by privilege resolution, or both. As a new feature, privileges that are inherited are stored in the user's dxrResolvedPrivilegesLink. In the Identity Manager's "Assigned by" column (and in WebCenter's "Mode" column), this is shown by the mode **inherited**. Note that this mode can be mixed with other modes that indicate the different types of sources for this privilege.

To populate the dxrResolvedPrivilegesLink once for all users, you need to run the privilege pesolution process for all users. To perform this task, change the subject filter to '(objectClass=dxrUser)' before the run.

5.8.4. Filtering Assigned Privileges

The Identity API contains a method listObjects in the Objects interface.

If assigned objects are requested (assignedRoles, assignedPermissions, assignedGroups, or assignedAccounts), the filter and base parameters are now considered. The intent of this step is to speed up the display of assignments to selected privileges in situations where users possess many privileges.

The following hints must be considered:

1. Set filter = null or base = null to get all assigned privileges.
2. If you want to read a certain assigned privilege, use the filter="(objectClass=**)" and the privilege's DN as the base. The filter "(objectClass=**)" is evaluated with best performance (always match). It is preferable to "(objectClass=dxrRole)" and so on.
3. Be aware that the assignments returned by the search may only be used for read/display. If you want to select an assignment and then update it later on, you need to search the assignments for the given type using filter = null or base = null and identify the assignments to be modified on this basis. The reason for this is that during save, all new assignments are compared to all old assignments. Assignments being read by a filter are ignored for update.
4. If you have already developed some code with Identity API, check the listObjects calls for assigned privileges. Set filter = null and base = null if you do not intend filtering here or if you want to update the assignments afterwards.

5.8.5. Migrating Role Parameter Access and Data

If a display attribute is defined for a role parameter (for example, cn), the value and the display value are stored and are also present in a request workflow.

The new role parameter format stores the value within the value tag. Thus, the extension is backward compatible to the old solution. If a special display value exists, it is stored in the key attribute of the value element, for example:

```
<roleparamvalues name="Project" uid="uid-7f001-cee271-fed935aa6d--
7eb4" type="DN">
    <value
key="OptimizeIT">cn=OptimizeIT,cn=Projects,cn=BusinessObjects,cn=My-
Company</value>
</roleparamvalues>
```

5.8.5.1. Migrating Java Code for Accessing Role Parameter Values

If you have programmed a Java algorithm to access role parameter values, you should read this section. Otherwise, you can ignore it.

Access to key and value is performed by the new KeyValue interface of the assignment

package:

```
public interface KeyValue {
	public void setKey(String key);
	public String getKey();
	public void setValue(String value);
	public String getValue();
}
```

If no display attribute is defined, the key remains empty and the data are provided by the value.

The RoleParameter interface has been adapted accordingly. The following four methods are affected by the change:

```
public interface RoleParameter {
	…
	public KeyValue[] getValue();
	public void setValue(KeyValue[] values);
	public String[] getStringValue();
	public void setValue(String[] values);
}
```

getValue() returns an array of KeyValue beans.

setValue() can be called with an array of KeyValue beans or with a String array. In the latter case, the strings are stored as values.

getStringValue() returns the keys if they exist, otherwise the values.

Access to role parameter handling in a request workflow job is described in the sample SampleJobForRoleParameterHandling.

In the following, migration to the new RoleParameter interface is described for this sample.

The following code must be changed:

1) In line 85: `String[] values = roleParam.getValue();`

The statement must be changed to
`String[] values = roleParam.getStringValue();`

2) In line 165: `DsmlModification[] modifications = rpMod.getValue();`

Note that DsmlModifcation must be replaced by KeyValueModification here (and in line 169).

3) In line 170: `String[] values = modification.getValue();`

This line must be replaced by the following two lines:

```
KeyValue[] kvs = modification.getValue();
String[] values = ParameterUtil.valuesFromKVs(kvs);
```

The first statement returns an array of KeyValue beans. The second converts it to an array of String, containing the keys if they exists, otherwise the values.

4) In line 203: `values = rpv.getValue();`

must be replaced by `values = rpv.getStringValue();`

5.8.5.2. Migrating Role Parameter Data in Assignments

If you have existing assignments with role parameters with display values, no migration is required for the data. The reason for this is that the display values are re-calculated automatically when the role parameter data are read in. This action is performed independent of whether a key attribute exists for the value. Thus, the displayed values are always up-to-date.



Saving the user will not change the format of the role parameter values unless a value has changed.

5.8.5.3. Migrating Role Parameter Data in Running Request Workflows

If you have running request workflows, no migration of the role parameter format is required, since the format is backward compatible.

5.8.6. Migrating Tcl-based Workflows to Handle Changes in the Object Search Operation

The **init.metacp** file includes the new Tcl variable **_md_req_attr_limit**, which was introduced to avoid passing huge attribute lists to the LDAP server. If more attributes than the given limit are defined, **metacp** internally requests all of the attributes from the LDAP server using the **-allattributes** switch in the **obj search** command. The default value for this variable is **64**.

The following problems may occur if the LDAP server requests all attributes:

- If the LDAP server recognizes alternate LDAP attribute names, it may return an attribute with a different LDAP attribute name than expected.
- Operational attributes are no longer returned.

Be aware of this behavior when you are running Tcl-based workflows and you find that some of the attributes are no longer synchronized. To solve the problem, you need to set the value of the Tcl variable ***md_req_attr_limit*** **to -1. For more information, see the _DirX Identity Troubleshooting Guide** and the *DirX Identity Meta Controller Reference*.

5.8.7. Migrating LDAP Realtime Workflows for Group Renaming

The **Join** section of the Group channel of the default LDAP realtime workflow has been changed and now looks as follows:

```
<joins xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core"
xmlns:spml="urn:oasis:names:tc:SPML:1:0">
    <join>
        <searchBase type="urn:oasis:names:tc:SPML:1:0#DN">
            <spml:id>${source.dxrPrimaryKeyOld}</spml:id>
        </searchBase>
    </join>
    <join>
        <searchBase type="urn:oasis:names:tc:SPML:1:0#DN">
            <spml:id>${source.dxrPrimaryKey}</spml:id>
        </searchBase>
    </join>
    <join>
        <filterExtension>
            <dsml:equalityMatch name="cn">
                <dsml:value>${source.cn}</dsml:value>
            </dsml:equalityMatch>
        </filterExtension>
    </join>
</joins>
```



The first join item using “${source.dxrPrimaryKeyOld}” is new. The change is necessary if a group has been renamed. In this case, the group is first searched with its old name. If it is found, it is updated and renamed in the connected system. Without this item, a new group is created in the connected system.



The LDAP object class **groupOfNames** doesn’t hold any attribute that never changes (for example, **employeeNumber** for LDAP object class **person)** and which therefore could be used when joining the entry. As a result, first joining with the old group name, then with current group name and finally with cn is performed.

So, if you copied the default workflow with older versions of DirX Identity, the new join item using “${source.dxrPrimaryKeyOld}” is missing in your LDAP realtime workflow. Thus, if you want group renaming to work in your environment, you should extend your LDAP workflow manually in the same way as the default LDAP realtime workflow. If you are sure that groups are never renamed in your environment, there is no need to extend the join

criteria.

5.8.8. Migrating Tcl-based and Realtime Workflows for New Object Classes dxrPersona, dxrFunctionalUser and dxrUserFacet

The DirX Identity installation provides a few (default) Tcl and realtime workflows that synchronize user objects. Up to now, these objects have been synchronized by either selecting the LDAP object class dxrUser or by using any other attribute (for example, employeeNumber) supposing that with such a filter a user object will match.

With DirX Identity V8.3A, you can use the new LDAP object classes **dxrPersona** (if you are interested in alternative representations for a user) and/or **dxrFunctionalUser** (if you are interested in resources that are assigned to a user). With DirX Identity V8.5, the new LDAP object class **dxrUserFacet** is available.

These new LDAP object classes are auxiliary object classes and are used in combination with the LDAP object class **dxrUser**. Therefore, when searching for dxrUser, you will also find objects with the LDAP object class dxrUser and dxrPersona, dxrUser and dxrFunctionalUser or dxrUser and dxrUserFacet (as long as the other filter attributes also match; but for the object class dxrPersona, the same set of attributes applies).

An update of the Tcl and realtime workflows is thus required, because when the join criteria results in a multiple match, none of the objects are synchronized. For export workflows, objects other than users would be exported, too.

5.8.8.1. Adapting Tcl (Import) Workflows

You must change the join expression of the output channel when running in MERGE mode. The following example shows how to exclude dxrPersona, dxrUserFacet and dxrFunctionalUser objects (pure LDAP filters must be used and note that the NOT filters can only be defined if you are using the “expert” filter; if you used the “table” format, you need to switch from “table” format to “expert” format):

Old join expression:

```
(&(employeeNumber=[lindex $rh_ldap(employeeNumber) 0])
       (objectClass=dxrUser))
```

New join expression:

```
(&(employeeNumber=[lindex $rh_ldap(employeeNumber) 0])
        (objectClass=dxrUser)
        (!(objectClass=dxrPersona))
        (!(objectClass=dxrUserFacet))
        (!(objectclass=dxrFunctionalUser)))
```

For details, see the channel Connectivity Configuration → Connected Directories → Default → Identity Store → Identity Store → Channels → LDIFFile2Ident.

5.8.8.2. Adapting Tcl (Export) Workflows

You must change the search filter in the input channel. The following example shows how to exclude dxrPersona, dxrUserFacet and dxrFunctionalUser objects (a GUI LDAP filter representation is shown):

Old search filter:

```
(l="munich" and objectClass="dxrUser")
```

New search filter:

```
(l="munich" and objectClass="dxrUser"
 and not (objectClass="dxrPersona")
 and not (objectClass="dxrUserFacet")
 and not (objectClass="dxrFunctionalUser"))
```

For details, see the channel Connectivity Configuration → Connected Directories → Default → Identity Store → Identity Store → Channels → Ident2LDIFFile.

5.8.8.3. Adapting Realtime Workflows

You must change the export search filter when reading from the LDAP directory (IdentityStore). You can use the same technique as described in “Adapting Tcl (Export) Workflows” to exclude “dxrPersona”, “dxrUserFacet" and “dxrFunctionalUser” objects in the realtime workflows. For details, see the channel Connectivity Configuration → Connected Directories → Default → Identity Store → Identity Store → Channels → LDIFFile → users.

You also need to change the Join expression when importing to the LDAP directory (IdentityStore). The following example shows how to exclude dxrPersona, dxrUserFacet and dxrFunctionalUser objects:

Old join expression:

```
<join>
     <filterExtension>
          <dsml:equalityMatch name="employeeNumber">
                    <dsml:value>${target.employeeNumber}</dsml:value>
          </dsml:equalityMatch>
      </filterExtension>
</join>
```

New join expression:

```
<join>
    <filterExtension>
        <dsml:and>
            <dsml:equalityMatch name="employeeNumber">
                 <dsml:value>${target.employeeNumber}</dsml:value>
            </dsml:equalityMatch>
            <dsml:equalityMatch name="objectClass">
                 <dsml:value>dxrUser</dsml:value>
            </dsml:equalityMatch>
            <dsml:not>
                 <dsml:equalityMatch name="objectClass">
                       <dsml:value>dxrPersona</dsml:value>
                 </dsml:equalityMatch>
            </dsml:not>
            <dsml:not>
                 <dsml:equalityMatch name="objectClass">
                       <dsml:value>dxrUserFacet</dsml:value>
                 </dsml:equalityMatch>
            </dsml:not>
            <dsml:not>
                 <dsml:equalityMatch name="objectClass">
                       <dsml:value>dxrFunctionalUser</dsml:value>
                 </dsml:equalityMatch>
            </dsml:not>
        </dsml:and>
    </filterExtension>
</join>
```

The changes listed above refer to “User Import” or “User Export” workflows. You also should keep in mind that other realtime workflows (synchronizing accounts to a connected system) might also be affected. The creation of new persona objects might result in the creation of new accounts for the relevant target systems (eventually with same set of attributes). Therefore, when the realtime workflows synchronize these accounts/personas to the connected system, you need to use a unique attribute in the Join operation (for example, **uid**).

Otherwise, the realtime workflow first synchronizes the account to the connected system. Later on, the persona object is synchronized, which results in a Rename operation in the connected system (for example, if the common attribute employeeNumber of the account and the persona object is used for joining).

5.8.9. Migrating the JMS Audit Configuration

The migration process extracts JMS Audit configuration values (user, password, url, queue name and logpath) that were hard-coded in XML server configuration (the old way up to V8.3) and stores them in specific attributes (the new way since V8.4).

Migration is done automatically during the first Configurator run. All workflows in the Connectivity database are processed.

You can start this migration step by hand at any time. It is located in:

install_path/**tools/migration/83**

As the Connectivity part is migrated, all arguments refer to the Connectivity configuration database.

Usage:

MigrateJMSauditConf.bat *host port user password ssl logfile*

Example:

```
MigrateJMSauditConf.bat localhost 389 "cn=admin,dxmC=DirXmetahub" dirx 0
trace.txt
```

5.9. Aspects Relevant for Upgrade from V8.2

This section describes all migration issues relevant to upgrading to the current version from DirX Identity V8.2.

5.9.1. Migrating Identity Server SSL Configurations

In V8.3, the setup and handling of SSL connections changed, including the distribution and location of key material.This section describes the necessary steps to migrate existing SSL key material.

Note that we strongly recommend setting up new key material, as the process has been enhanced to use a CA certificate, which makes distribution of key material much easier.It also has been decided to support only server-side SSL.Finally, you can only secure all Identity server connections at once, not separately.

If you want to use your previously created SSL key material, you need to decide if you'll use the key material from the former Java-based Server, from the secured SOAP port or from the messaging service from the C++-based Server.

5.9.1.1. Using the Java-based Server Key Material

If you previously used SSL connections to the Java-based Server, you are using the following files in the folder:

- *install_path*/**ids-j-***domain*-**S***n*/**private/server.crt** (certificate of the server)

- *install_path*/**ids-j-***domain*-**S***n*/**private/server-keystore** (private-key/certificate of the server)
- *install_path*/**ids-j-***domain*-**S***n*/**private/server-truststore**

Perform the following steps:

- Copy the files to the new location *install_path**/ssl*. Adjust the keystore and truststore passwords in the file *install_path*/**ssl/password.properties** accordingly.
- Import the **server.crt** into the JRE's certificate truststore. The default truststore is the JRE's cacerts file *jre_root*/**jre/lib/security/cacerts.** To import the **server.crt** file, use the DirX Identity Manager. In the **Tools** menu, choose **Options**. On the next page, the Manager's truststore is selected by default ("This application's installation folder" is already selected and the file *install_path*/**GUI**/**cacerts** is shown). Choose **Java Runtime Environment**. Choose **Import** and then select the server certificate you want to import. When prompted for the keystore password, enter the default value **changeit**. Click **OK** and the certificate will be imported.
- Convert server-keystore from JKS format to PKCS#12 format and then to PEM format. To perform this task, see the script *install_path**/ssl/convert_ServerKeystoreToPem.bat*. The best approach is to copy the file and then adjust the variables. Note that you must use your **server.crt** instead of an input file **ca.crt.**

You should now have the following files:

- **server.crt**
- **server-keystore**
- **client-truststore**
- **server-key.pem**
- **ca-crt.pem** (Containing your **server.crt** certificate)

If you have additional Java-based Servers on the same or other machines, you must add each **server.crt** to other **client-truststore** and **ca-crt.pem** files. Consult the documentation from Oracle for the **keytool** tool and from openSSL for the **openssl** tool.

5.9.1.2. Using the Messaging Service Key Material

If you previously used SSL connections to the messaging service in the C++-based Server, you are using the following files in the folder:

- *install_path*/**security/certificates/qm.crt.p12** (private key/certificate of the server)
- *install_path*/**security/certificates/qm.crt** (certificate of the server)
- *install_path*/**security/java/truststore** (truststore with certificate)

Perform the following steps:

- Copy the files to the new location *install_path**/ssl*. Rename qm.crt to server.crt and truststore to client-truststore. Adjust the keystore and truststore passwords in the file *install_path**/ssl/password.properties* accordingly.

- Import the server.crt into the JRE's certificate truststore. The default truststore is the JRE's cacerts file *jre_root*/**jre/lib/security/cacerts.** To import the **server.crt** file, use the DirX Identity Manager. In the **Tools** menu, choose **Options**. On the next page, the Manager's truststore is selected by default ("This application's installation folder" is already selected and the file *install_path*/**GUI/cacerts** is shown). Choose **Java Runtime Environment**. Choose **Import** and then select the server certificate you want to import. When prompted for the keystore password, enter the default value **changeit**. Click **OK** and the certificate will be imported.
- Convert the PKCS#12 format to JKS format. Use the **keytool** utility (PASSWORD_P12 is the old password, NEW_PASSWORD the new one):

```
keytool -importkeystore -srckeystore qm.crt.p12 -srcstoretype pkcs12
-srcstorepass PASSWORD_P12 -srcalias messageserver -destkeystore server-
keystore -deststoretype jks -deststorepass NEW___PASSWORD -destalias
identity-server
```

- Convert the PKCS#12 format to the PEM format:

```
openssl pkcs12 -in qm.crt.p12 -out server-key.pem -passin
pass:_PASSWORD_P12_ -passout pass:_PASSWORD_NEW_

openssl x509 -inform DES -in qm.crt –outform PEM -out ca-crt.pem
```

You should now have the following files:

- **server.crt**
- **server-keystore**
- **client-truststore**
- **server-key.pem**
- **ca-crt.pem** (Containing your **server.crt** certificate)

5.9.1.3. Using the Key Material from the C++-based Server SOAP Port

If you previously used SSL connections to the SOAP service in the C++-based Server, you are using the following files in the folder:

- **svr.pem** (private key/certificate of the server)
- **cacert.pem** (certificate of the server)

Perform the following steps:

- Copy the files to the new location *install_path**/ssl*. Rename **cacert.pem** to **ca-crt.pem** and **svr.pem** to **server-key.pem**. Adjust the keystore and truststore passwords in the file *install_path**/ssl/password.properties* accordingly.
- Convert PEM format to crt (DER) format:

```
openssl x509 -outform der -in ca-crt.pem –out server.crt
```

```
keytool -import -v -alias <your alias> -file <your file>.pem -keystore <your
key store>.jks -storepass <your storepass>
```

- Import **server.crt** into the JRE's certificate truststore. The default truststore is the JRE's **cacerts** file *jre_root/**jre/lib/security/cacerts.* To import the **server.crt** file, use the DirX Identity Manager. In the **Tools** menu, choose **Options**. On the next page, the Manager's truststore is selected by default ("This application's installation folder" is already selected and the file *install_path**/GUI/cacerts* is shown). Choose **Java Runtime Environment**. Choose **Import** and then select the server certificate you want to import. When prompted for the keystore password, enter the default value **changeit**. Click **OK** and the certificate will be imported.
- Convert the PKCS#12 format to JKS format. Use the **keytool** utility (PASSWORD_P12 is the old password, NEW_PASSWORD the new one):

```
keytool -importkeystore -srckeystore qm.crt.p12 -srcstoretype pkcs12
-srcstorepass PASSWORD_P12 -srcalias messageserver -destkeystore server-
keystore -deststoretype jks -deststorepass NEW___PASSWORD -destalias
identity-server
```

- Convert the PKCS#12 format to PEM format:

```
openssl pkcs12 -in qm.crt.p12 -out server-key.pem -passin
pass:_PASSWORD_P12_ -passout pass:_PASSWORD_NEW_

openssl x509 -inform DES -in server.crt –outform PEM -out ca-crt.pem
```

- Insert the certificate into the client truststore:

```
keytool -import -alias identity-server -keystore clientstore -file
server.crt
```

You should now have the following files:

- **server.crt**
- **server-keystore**
- **client-truststore**
- **server-key.pem**
- **ca-crt.pem** (Containing your **server.crt** certificate)

5.9.2. Cleanup regarding Worker Containers

DirX Identity Java-based worker containers have been supported since DirX Identity V8.2, but they are no longer supported with this version.

If your installation is an upgrade installation of one of these DirX Identity versions, you need to perform the manual cleanup steps regarding worker containers described in this section. These steps are platform-dependent and include:

- Service configuration cleanup
- File system cleanup

5.9.2.1. Windows Platforms

No steps are required regarding service configuration cleanup.

No steps are required regarding *install_path***/configuration.ini.** Just be aware that these properties are no longer relevant for this version:

- **ConfiguredComponents.IdS-J-Worker**
- **InstalledComponents.IdS-J-Worker**
- All properties starting with **IdS-J.w**.
- **option.idsj_worker**

To clean up the file system, perform this step:

- For each subfolder *idsj* where the name is either **ids-j** or starting with **ids-j-**, remove the folder *install_path*/*idsj*/**worker** recursively.

5.9.2.2. Linux Platforms

No steps are required regarding service configuration cleanup. As for Windows platforms, no steps are required regarding *install_path*/configuration.ini*.

To clean up the file system, perform these steps:

- For each subfolder *idsj* where the name is either **ids-j** or starts with **ids-j-**, remove the folder *install_path*/*idsj*/**worker** recursively.
- Remove the files **S99dmsvrwo-*** in the folder *install_path*/**etc**.

5.9.3. Cleanup regarding JAXB-API and JAXWS-API in Tomcat

During Web Center deployment, Web Center for Password Management, Web Center for SAP NetWeaver, Provisioning Web Services of DirX Identity V8.2, the following files are created in *tomcat_install_path*/endorsed*:

- jaxb-api.jar
- jaxws-api.jar

These files are obsolete because the implementation of these standards is now part of Java Runtime >= 1.8. Remove these files if you can ensure that they are related to one of the deployments listed above and if you can exclude negative side effects for any other application you deployed into that Tomcat.

5.9.4. Adapting Custom Scripts

If you do not have any custom scripts that use **java** or **keytool** and which originate from DirX Identity V8.2, you can skip this section. Otherwise, you need to adapt these scripts as

described in this section.

DirX Identity scripts that use **java** or **keytool** have been modified so that they are correct regarding the following issues:

1. **Using the correct JRE location.** As described in the *DirX Identity Installation Guide*, the location *dxi_java_home* of the JRE for DirX Identity is defined during installation. When selecting a customer-supplied JRE as the JRE for DirX Identity, the JRE in *install_path**/lib/jre* no longer exists. In this case, scripts using **java** or **keytool** from the location *install_path**/lib/jre* will no longer work. For this reason, DirX Identity scripts which use **java** and **keytool** use the environment variable **DXI_JAVA_HOME** for referencing the correct JRE location.

5.9.4.1. Adapting Java Calls in Windows Batch Files

To adapt the **java** calls in Windows batch files:

- Insert the call **"%DIRXIDENTITY_INST_PATH%/setdxienv.bat"** prior to calling **java**.
- Adapt the **java** call: **"%DXI_JAVA_HOME%/bin/java"** (or alternatively: set **PATH=%DXI_JAVA_HOME%/bin;%PATH%** followed by simply calling **java**).
- Test your adaptation.

5.9.4.2. Adapting keytool Calls in Windows Batch Files

To adapt the **keytool** calls in Windows batch files:

- Add the call **"%DIRXIDENTITY_INST_PATH%/setdxienv.bat"** prior to calling the program.
- Adapt the **keytool** call: **"%DXI_JAVA_HOME%/bin/keytool"** (or alternatively: set **PATH=%DXI_JAVA_HOME%/bin;%PATH%** followed by simply calling **keytool**).
- Test your adaptation.

5.9.4.3. Adapting Java Calls in UNIX Shell Scripts

To adapt the **java** calls in UNIX shell scripts:

- Adapt the java call: **"$DXI_JAVA_HOME/bin/java"** (or alternatively: **PATH=$DXI_JAVA_HOME/bin:$PATH** followed by simply calling **java**).
- Test your adaptation.

5.9.4.4. Adapting keytool Calls in UNIX Shell Scripts

To adapt the **keytool** calls in UNIX shell scripts:

- Adapt the **keytool** call: **"$DXI_JAVA_HOME/bin/keytool"** (or alternatively: **PATH=$DXI_JAVA_HOME/bin:$PATH** followed by simply calling keytool).
- Test your adaptation.

5.9.4.5. Adapting Java Calls in Tcl scripts

After installation and configuration, the suitable Java location is stored in *install_path* /*basic.input.tcl*.Perform these adaptations:

- Modify the definition of the Java path and suitable computation.
- Test your adaptation.

5.10. Aspects Relevant for Upgrade from V8.2C

This section describes all migration issues relevant to upgrading to the current version from DirX Identity V8.2C.

5.10.1. Deleting the jms.jar File

The **jms.jar** file is no longer provided with the installation.JMS messaging is now provided by **geronimo-jms_1.1_spec.jar** (which is packaged with ActiveMQ and integrated into the DirX Identity installation).

As a result, if you made an upgrade installation, you must delete **jms.jar** in your installation directory manually.The following file needs to be deleted:

*install_path***/web/webCenter-***domain***/webCenter/WEB-INF/lib/jms.jar**

5.10.2. Migrating Realtime Channels to Support Realtime Delta Workflows

The channel's XML definition (LDAP attribute **dxmContent**) needs to be updated to support the realtime delta workflow feature.

The Configurator performs this migration automatically when it is run for the first time and processes all workflows in the Connectivity database.

You can start this migration step by hand at any time. It is located in:

install_path/**tools/migration/82C/rtworkflows**

As the Connectivity part is migrated, all arguments refer to the Connectivity configuration database.

Usage:

MigrateDeltaWorkflow.bat *host port user password ssl logfile*

Example:

```
MigrateDeltaWorkflow.bat localhost 389 "cn=admin,dxmc=dirxmetahub" dirx 0
log.txt
```

5.10.3. Updating the Realtime Event Port

The XML definition of the event port in realtime workflows (LDAP attribute **dxmContent**) contains a “Connection” section that uses invalid references to non-existing LDAP objects. As this section is not evaluated by the realtime workflows, a migration procedure will remove it.

The Configurator performs this migration step automatically when it is run for the first time and processes all workflows in the Connectivity database.

You can start this migration step by hand at any time. It is located in:

install_path/**tools/migration/82C/rtworkflows**

As the Connectivity part is migrated, all arguments refer to the Connectivity configuration database.

Usage:

MigratePubConnector.bat *host port user password ssl logfile*

Example:

```
MigratePubConnector.bat localhost 389 "cn=admin,dxmc=dirxmetahub" dirx 0
log.txt
```

5.10.4. Handling Minimum Source Entries in Tcl-based Workflows

No migration for this parameter is required. This section is just to inform you about some minor changes:

The “Minimum Source Entries” parameter is now also evaluated if paging mode is turned on. The parameter is evaluated both for export and import synchronization. In addition, new entries were added and existing ones were modified in any case in old releases (without paging). The minimum number of entries was checked only when entries needed to be deleted, and deletions were not made if the minimum number of source entries was too small. With the new release, the minimum number of source entries is checked first, and synchronization starts only if the minimum number is OK.

5.10.5. Creating the dxrUid Attribute in the Provisioning Tree

For audit purposes, almost all objects in the Provisioning tree are required to hold the attribute **dxrUid**. Consequently, a migration tool checks the Provisioning tree and creates the attribute dxrUid where missing.

The migration tool creates the dxrUid attribute (if missing) under the following conditions:

- One of the object class values begins with the prefix **dxr**
- One of the object classes is dxmComponentDescription
- One of the object classes is dxmIDMWorkflowDefinition

- One of the object classes is dxmIDMActivityDefinition
- One of the object classes is dxmIDMNestedActivityDefinition
- One of the object classes is dxmEscalationDefinition

Objects that have different object classes than the ones listed above are not updated (because the dxrUid attribute is not defined for these object classes).

The Configurator performs this migration step automatically during its first run.

You can start this migration step by hand at any time. It is located in:

install_path/**tools/migration/82C/database**

As the Provisioning part is migrated, all arguments refer to the Provisioning configuration database.

Usage:

MigrateUid.bat *host port user password ssl domain-DN logfile*

Example:

```
MigrateUid.bat localhost 389 "cn=DomainAdmin,cn=My-Company" dirx 0 "cn=My-Company" log.txt
```

Note: due to the new paging features in DirX V8.2, a sizelimit error may occur and the tool might not be able to read all requested entries. In this case, simply start the tool again until only those entries remain whose object class does not allow setting a dxrUid (for example, dxmObjectCollection).

5.10.6. Migrating the e-Mail Body in Reject e-Mails of Request Workflows

If the feature **Reduce Runtime Activities** is set in **Approval activities**, there are many approvers assigned to one activity.

Consequently, when using the token **${activity.participantEntries}** in the e-mail body, all approvers are shown as the ones that rejected. Thus, a new token must be used that shows only the one(s) that really rejected.

Note that there is no automatic migration of the e-mail body. Therefore the following change needs to be made manually; for example, in the “Notification If Rejected” part of the 4-Eye Approval workflow of My-Company domain:

Here is the old e-mail body:

```
The assignment of privilege ${workflow.resources[0].dxrassignto@cn}
to user ${workflow.subject.cn} was rejected.
For questions about this decision, please contact the persons that
```

```
rejected the request:
<? for activity in ${workflow.activities} ?>
    <? if ${activity.approvalResult} == "REJECTED" ?>
        <? for participant in ${activity.participantEntries} ?>
           - Activity step: '${activity.name}': User
${participant.cn} with reason: ${activity.reason}
        <? endfor ?>
    <? endif ?>
<? endfor ?>
…
```

Here is the new e-mail body:

```
The assignment of privilege ${workflow.resources[0].dxrassignto@cn}
to user ${workflow.subject.cn} was rejected.
For questions about this decision, please contact the persons that
rejected the request:
<? for activity in ${workflow.activities} ?>
    <? if ${activity.approvalResult} == "REJECTED" ?>
        <? for participant in ${activity.approvers} ?>
             - Activity step: '${activity.name}': User
${participant.cn} with reason: ${activity.reason}
        <? endfor ?>
    <? endif ?>
<? endfor ?>
```

5.10.7. Migrating the URL of the Source Ticketing Sample Web Service

The URL of the Source Ticketing sample Web Service shipped with the new product release has changed. Therefore, applications which use the Web Service of the new product release must be changed so that they use a URL of the form **http://***host:*port**/sourceTicketing/sourceTicketingService** (instead of **http://***host:*port**/spmlsoapservice/services/SpmlSoapService**). For details about this topic, see the subsection "Preparing the Web Service Environment" in the section "Working with Source Tickets" of the chapter "Follow-up Tutorials" in the *DirX Identity Tutorial*.

5.10.8. Migrating SPML Filters that Use Wildcards

The conversion of an SPML filter expression containing a wildcard (*) character to an LDAP filter expression by the LDAP connector has changed with product versions newer than V8.2C.

This kind of filter is now converted according to the standard OASIS SPML filter

specification. This change means that a wildcard contained in a value of an SPML filter expression, like

```
<spml:filter>
     <dsml:equalityMatch name="sn">
         <dsml:value>a*</dsml:value>
     </dsml:equalityMatch>
</spml:filter>
```

is now interpreted as part of the value and not as a wildcard. As a result, it is escaped in the converted LDAP filter (“sn=a/**”) passed to the LDAP server with the consequence that users with a surname equal to *a*** are searched for, not the users with a surname beginning with **a**.

If wildcard searches are intended, the SPML filter element substring together with the elements initial, any or final must be used. The following snippet defines the SPML filter to handle the example of searching for all users with surname beginning with **a**:

```
<spml:filter>
     <dsml:substrings name="sn">
         <dsml:initial>a</dsml:initial>
     </dsml:substrings>
</spml:filter>
```

If customers, for example, use their own workflow definitions with wildcards specified in, for example, channel search filters, they must adapt their definitions accordingly.

5.11. Aspects Relevant for Upgrade from V8.2B

This section describes all migration issues relevant to upgrading to the current version from DirX Identity V8.2B.

5.11.1. Updating Manager Profiles for the Data View

In V8.2B, the display of Data View search result entries was wrong (like the Provisioning View and not the Data View) when you used server=”DirXmetaRole” for the quick search panel.The workaround was to delete/rename the server attribute from the ldapquicksearchpanel definition.If you want to use the server=”DirXmetaRole” (you don’t have to select a server during Data View search):

- Activate in *install_path*/**GUI/profiles/dxrdataView.xml**:

  ```
  <ldapquicksearchpanel displayName="Search" showAdvancedButton="true"
  useProfiles="true" server="DirXmetaRole" />
  ```

- Add filterServer="raw" in *install_path*/**GUI/profiles/dxdViewGroup.xml**:

```
<viewgroup name="dataview" filterServer="raw" displayName="Data View">
```

5.11.2. Migrating JMS Subscriptions

Subscriptions for C++-based Server were migrated automatically during upgrade installation and configuration. After you complete the upgrade, check that only subscriptions in the new format exist. This format is described in the section “Issues Relevant for Upgrade from 8.2B” → “Migration of JMS Subscriptions” in the “Introduction” chapter. If you find subscriptions in the old format, delete them.

Hint: On the host of the C++-based Server, you’ll find **migration.ini** or **migration.tmp.ini** in *install_path***/ats/msgstore**. In this file, you’ll find the old subscriptions (on the right side and the new subscriptions on the left side of the equals sign).

5.11.3. Migrating the Source Ticketing Sample Web Service

See the subsection "Migrating the URL of the Source Ticketing Sample Web Service" in the section "Aspects Relevant for Upgrade from 8.2C".

5.11.4. Migrating Request Workflow Parameters to be Stored as dxmSpecificAttributes

Many request workflow parameters were stored in the attribute “dxmContent” (in XML format). For better performance and to be able to search for these parameters, these parameters are now stored in the attribute “dxmSpecifiAttriibutes”. For example:

dxmSpecificAttributes=escalationLevel 1

The Configurator performs this migration step automatically when it runs for the first time and processes all workflows in the Provisioning database.

You can start this migration step by hand at any time. It is located in:

install_path/**tools/migration/**82B/reqworkflows

As the Provisioning part is migrated, all arguments refer to the Provisioning configuration database.

Usage:

MigrateReqWFldapAttrs.bat *host port user password ssl logfile*

Example:

```
MigrateReqWFldapAttrs.bat localhost 389
```

"cn=DomainAdmin,cn=My-Company" dirx 0 "cn=My-Company" trace.txt

5.12. Aspects Relevant for Upgrade from V8.2A

This section describes all migration issues relevant to upgrading to the current version from DirX Identity V8.2A.

5.12.1. Adapting Object Descriptions

The behavior of properties of type “siemens.dxm.util.GeneralizedTime” without an explicit editor definition has changed.In earlier versions, only the date part of these properties was displayed.Now both the date and time part are shown.If only the date should be displayed, you need to insert an explicit editor definition.The relevant value is:

“siemens.DirXjdiscover.api.ldap.beans.JnbLdapGeneralizedDate”

5.12.2. Updating the SSL Flag for Messaging

The SSL flag for messaging is no longer located at the Configuration → Messaging Services → Message Server object. It has been shifted to the corresponding service object.

If you used SSL for messaging, perform the following steps:

- Start the DirX Identity Manager.
- Log in into Connectivity view group.
- Click the Design Mode button (Tool bar).
- Open **Configuration** → **Services** → **System** → **Message Service** object.
- Click **Edit** and set the **SSL** flag at this object.
- Set the secure port with the correct information. Enable this field with the first flag.
- Click **Save**.
- Disable Design Mode.
- Check that both the **SSL** flag and the **Secure Port** are visible and set correctly.

5.12.3. Solving Class Loading Problems in IdS-J Server

In previous releases of DirX Identity, the default location for customer-specific jar files concerning

- Own Java mappings, user hooks, connector filters for realtime workflows
- Custom connectors used in realtime workflows

was

install_path/**ids-j**-*domain*-**S***n*/**confdb/jobs/framework/lib**

This location has changed and now the jar files are located in the following folder:

install_path/**ids-j**-*domain*-**S***n*/**confdb/common/lib**

Make sure that the jar files are removed from the old location.

5.12.4. Adjusting SAP R/3 UM Workflows

SAP R3 UM agent/connector: A new flag **directlyAssignedRolesOnly** (set to **true**) allows export of only directly assigned roles. The default value is **false**. If you want to adjust your existing workflows perform the following steps:

1. Tcl-based Workflows: Job SAPR3UM2Ident_XXXAccount_UMAgent file Configuration.xml:

```
 <connector
        role="connector"
        className="siemens.dxm.connector.sapUM.sapUMuser"
        …>
        <connection type="SAP_R3_UM" ...>
             ...
            <property name="directlyAssigndRolesOnly"
value="true"/>
        </connection>
```

2. Realtime workflows: ts = Target System Port content tab of join activity:

```
<port connector="SapR3UMConnector" ...
        <connector className="siemens.dxm.connector.sapUM.sapUM"
mode="inout" ...>
                <connection password=...
                    ...
                    <property name="directlyAssigndRolesOnly"
value="true"/>
                </connection>
```

5.12.5. Fixing the ClearOnMasterRemoval Typo

In **UserCommon.xml**, this flag was not correctly written.

For each domain, navigate to Provisioning view group → **Domain Configuration** → **Customer Extensions** → **Object Descriptions** → **UserCommon.xml**

Replace each occurrence of the string ClearOnMasterRemova=" with ClearOnMasterRemoval=".

5.12.6. Using the New Identity API

The Identity API has changed. For details, see the DirX Identity API documentation delivered with Web Center.

5.12.7. Adapting Policies

The object description name for a Provisioning rule had to be changed from ProvisionRule to dxrProvisionRule and the object description name for password policies had to be changed from dxrPwdPolicyCheck to dxrPwdPolicy.

If attribute polices, event policies or delete policies for these types exist in your environment, adapt these policies accordingly, or they will no longer work.

5.12.8. Migrating the Source Ticketing Sample Web Service

See the subsection "Migrating the URL of the Source Ticketing Sample Web Service" in the section "Aspects Relevant for Upgrade from 8.2C".

6. Known Issues

6.1. Overwritten cert8.db file during Update installation

During an update or upgrade installation the cert8.db file is written again.If you used this default file for your certificates they are lost.

To prevent such type of errors, use another filename for your certificates and set the corresponding environment variable.

6.2. Overwritten agent batch files during Update installation

During the DirX Identity installation the agent batch files in the bin directory of the installation path are overwritten.If you changed something in these files (for example more memory for the Java processes) this information is lost.

To avoid this problem, you should copy the agent batch file before you perform the changes.To use the copied and changed files, set the link from the Agent objects in the DirX Identity configuration database (Connectivity Configuration Data → Agents) to the new files.

In a distributed environment be sure that all DirX Identity servers where this agent shall run have a copy of the changed file.

6.3. Deletion of Old Objects

The configuration runs a set of scripts during an upgrade installation to delete superfluous objects.

With the InitialConfiguration step “Connectivity Schema and Data Configuration” the outdated objects listed in the files containing “Hub” in the name are deleted in the Connectivity configuration.With the step “Domain configuration” the outdated objects of the specific domain listed in the files containing “Role” in the name are deleted in the Provisioning configuration.

If you have added parts of the deleted objects for example in the context of a collection import, you can either run the scripts again as described below or just delete the objects listed in the files with the Identity Manager.

These files with objects to be deleted exist:

- **DeleteOutdatedSubtreesHub_82A.txt**
  if you upgrade from 8.2A to 8.2B.
- **DeleteOutdatedSubtreesRole_82A.txt**
  if you upgrade from 8.2A to 8.2B.

- **DeleteOutdatedSubtreesSampleDomain_82A.txt**
  if you upgrade from 8.2A to 8.2B.
- **DeleteOutdatedObjectsHub_82B.txt**
  if you upgrade from 8.2B to 8.2C.
- **DeleteOutdatedSubtreesHub_82B.txt**
  if you upgrade from 8.2B to 8.2C.

If you have to delete the objects again after the InitialConfiguration run, which can only happen in the context of a collection import or other ldif file import after the InitialConfiguration, you can run the scripts again.

To run the scripts for deletion of objects listed in ***.ldif** files:

Open a dos prompt,

change to the directory *install_path***\confdb\data**,

call **..\..\bin\metacp ..\tcl\load.ldif.change.tcl** *OutdatedObjects_filename trace_filename*

To run the scripts for deletion of objects listed in *.txt files:

Open a dos prompt,

change to the directory *install_path*\confdb\data*,

call **..\..\bin\metacp ..\tcl\removeSubtree.tcl** *user password server port ssl subtree_dn tracefile* for every subtree dn listed in the ***.txt** file.

Because the **removeSubtree.tcl** script must be called for every subtree it is easier to delete the subtrees with the Identity Manager.

6.4. UID Generation Fails

Running migration.bat from the folder

`install-path\GUI\migration\CreateDxrUids`

can result in the

`Set uid value for object dn failed.`

Cause could be that you extended your object description for example with save scripts. To solve this problem be sure that the DirX Identity Manager and the migration script use the same ClassPath.

6.5. Sample Domain: Doubled Memberships (ADD/OK)

Updating an existing and used sample domain can result in doubled memberships at

accounts and groups. That means the dxrMemberADD and dxrMemberOK are both filled with the same members.

This problem comes from the fact that initially the members are all in ADD state. If you played with the sample domain, some or all of the members go to OK state. Updating the sample domain during an upgrade adds all ADD states again.

You can easily correct this problem if you resolve all users of the My-Company domain.

Run the Privilege resolution with the filter (objectClass=dxrUser). This resolves all users and corrects these doubled memberships.

6.6. Inconsistent Object Descriptions (single / multi value)

If an attribute is filled in the LDAP directory with multiple values and the object description for this attribute is defined as single value, errors could occur that were hard to find.

Starting with DirX Identity V8.2B such inconsistencies are reported via a warning message:

WAR(STG617): Multiple values exist for single value property '<attributename>' of dn=...

If you encounter such a warning, check the corresponding object description of that object and correct it.

7. Migrating JavaScript from Rhino to GraalVM

DirX Identity V8.10 supports both Rhino and GraalVM JavaScript (GraalJS). Where the script definition supports engine selection, you can migrate one script at a time and leave other scripts on Rhino. Changing the engine does not rewrite the script.

7.1. Before You Migrate

- Use a DirX Identity build that supports the GraalVM executor and the Java runtime supported by that build. See Update Tomcat and Java Environment. Selecting the GraalVM executor is a script configuration change, not a change to the JVM that runs DirX Identity.
- Inventory custom JavaScript and its callers. Include inline XML scripts, references to JavaScript entries, and provisioning rule actions. Record the current engine setting, the expected result type, and any changes the script makes to DirX Identity objects.
- Back up both the script source and its configuration. In a test environment, capture the Rhino behavior with representative inputs, including empty values and error cases.

7.2. Selecting the JavaScript Executor

In an XML object description, set the `language` attribute on the `<script>` element that defines or references the script. Use exactly `js-rhino` to select Rhino or `js-graalvm` to select GraalVM. The setting is per script definition, not per JavaScript file. The `scriptContext` variable still provides access to the application context; do not set the language inside the script body.

For example, the following are two versions of an inline property default script. Use only one definition at a time:

```
<script name="defaultDescription" language="js-rhino" return=
"result">
  <![CDATA[
    var result = "New account";
  ]]>
</script>
```

```
<script name="defaultDescription" language="js-graalvm"
return="result">
  <![CDATA[
    var result = "New account";
  ]]>
```

```
</script>
```

For a referenced script, add `language` to the same `<script>` element as the `reference` attribute. If several definitions reference the same source, check each one before editing the shared source. Use separate entries during the transition if the Rhino and GraalVM versions of the code differ.

For a provisioning rule action implemented as a JavaScript operation, set the operation object's `dxrType` attribute to `javascript/js-rhino` or `javascript/js-graalvm`. The rule runner reads the script's result from the global `success` variable; keep that variable when migrating the action.

In V8.10, these definitions use Rhino if no engine is specified. An unrecognized language value logs a warning and falls back to the default. Set the engine explicitly for scripts under migration so a test does not accidentally run on Rhino. Other scripting features may choose the engine differently; do not assume that they accept the XML attribute or operation type described here.

7.3. Adapting JavaScript Code

7.3.1. Import Java Classes Explicitly

Replace Rhino-specific imports such as `importPackage()` and `importClass()` with `Java.type()` and a fully qualified class name. Do not rely on implicit package globals or enable compatibility modes to avoid porting the code.

Rhino

```
importPackage(java.util);
var names = new ArrayList();
names.add("example");
```

GraalVM

```
var ArrayList = Java.type("java.util.ArrayList");
var names = new ArrayList();
names.add("example");
```

7.3.2. Check Strings, Java Calls, and Result Types

- Use `value.length` rather than `value.length()` for strings in GraalVM, including strings returned by Java. Check for `null` or `undefined` first if the property is optional.
- Where Java methods are overloaded, verify which overload receives your arguments. If necessary, select it explicitly, for example `names["remove(int)"](0)` for index-based removal from a Java list.

- Keep the caller's result contract. If a definition specifies `return="result"`, assign a top-level `var result` on every relevant code path. Do not replace it with a function-local variable, a top-level `let` or `const`, or an assumption about the script's last expression.
- Distinguish JavaScript arrays from Java arrays and collections. Verify the type required by the consumer before adding conversions. For example, V8.10.16 improves conversion to Java `Object[]` for assigned-group filters in `TSAccounts.xml`; test `Groups.Assigned` with no groups and with multiple groups.
- Review Rhino-specific exception handling and implicit type coercion. Test both successful and failing Java calls rather than assuming Rhino exception objects behave the same way in GraalVM.

7.4. Rolling Out and Troubleshooting

1. Start with a low-risk script. Port its Java interoperability code, select GraalVM in the configuration, and apply the updated configuration to the component that runs the script.
2. Exercise the script through its real caller, such as a property edit in DirX Identity Manager or Web Center, or a provisioning rule evaluation. Compare returned values and types, saved attributes, and rule outcomes with the Rhino baseline. Repeat execution with different objects to check that the script does not depend on state from a previous invocation.
3. If the script fails, check the executing component's logs for the script name, error message, and source location when available:
    - If the engine setting appears to have no effect, check the exact selector value and confirm the component loaded the updated definition.
    - If a class or import is not found, check the `Java.type()` class name and that the required library is available to the executing component.
    - If the result is missing or rejected, check the configured result variable, its scope and value on each path, and the type the caller expects.
4. Deploy validated scripts in small batches. If a script has to be rolled back, restore both its previous source and engine setting, then reapply the configuration. Switching to Rhino alone can leave it running code written for GraalVM.

For more Java interoperability examples, see the GraalJS Migration Guide from Rhino to GraalJS and Java Interoperability guide.

Further Useful Links for GraalVM Migration

https://github.com/oracle/graaljs/blob/master/docs/user/RhinoMigrationGuide.md
https://github.com/oracle/graaljs/blob/master/docs/user/NashornMigrationGuide.md
https://github.com/oracle/graaljs/blob/master/docs/user/JavaScriptCompatibility.md
https://github.com/oracle/graaljs/blob/master/docs/user/FAQ.md

DirX Product Suite

The DirX product suite provides the basis for fully integrated identity and access management; it includes the following products, which can be ordered separately.



DirX Identity

DirX Identity provides a comprehensive, process-driven, customizable, cloud-enabled, scalable, and highly available identity management solution for businesses and organizations. It provides overarching, risk-based identity and access governance functionality seamlessly integrated with automated provisioning. Functionality includes lifecycle management for users and roles, cross-platform and rule-based real-time provisioning, web-based self-service functions for users, delegated administration, request workflows, access certification, password management, metadirectory as well as auditing and reporting functionality.



DirX Directory

DirX Directory provides a standards-compliant, high-performance, highly available, highly reliable, highly scalable, and secure LDAP and X.500 Directory Server and LDAP Proxy with very high linear scalability. DirX Directory can serve as an identity store for employees, customers, partners, subscribers, and other IoT entities. It can also serve as a provisioning, access management and metadirectory repository, to provide a single point of access to the information within disparate and heterogeneous directories available in an enterprise network or cloud environment for user management and provisioning.



DirX Access

DirX Access is a comprehensive, cloud-ready, scalable, and highly available access management solution providing policy- and risk-based authentication, authorization based on XACML and federation for Web applications and services. DirX Access delivers single sign-on, versatile authentication including FIDO, identity federation based on SAML, OAuth and OpenID Connect, just-in-time provisioning, entitlement management and policy enforcement for applications and services in the cloud or on-premises.



DirX Audit

DirX Audit provides auditors, security compliance officers and audit administrators with analytical insight and transparency for identity and access. Based on historical identity data and recorded events from the identity and access management processes, DirX Audit allows answering the “what, when, where, who and why” questions of user access and entitlements. DirX Audit features historical views and reports on identity data, a graphical dashboard with drill-down into individual events, an analysis view for filtering, evaluating, correlating, and reviewing of identity-related events and job management for report generation.

For more information: support.dirx.solutions/about

Eviden is a registered trademark © Copyright 2026, Atos Group – All rights reserved.

Legal remarks

On the account of certain regional limitations of sales rights and service availability, we cannot guarantee that all products included in this document are available through the Eviden sales organization worldwide. Availability and packaging may vary by country and is subject to change without prior notice. Some/All of the features and products described herein may not be available locally. The information in this document contains general technical descriptions of specifications and options as well as standard and optional features which do not always have to be present in individual cases. Eviden reserves the right to modify the design, packaging, specifications and options described herein without prior notice. Please contact your local Eviden sales representative for the most current information. Note: Any technical data contained in this document may vary within defined tolerances. Original images always lose a certain amount of detail when reproduced.