SAP NetWeaver® Identity Management
Identity Center

Implementation guide
- Staging environment

Version 7.1 Rev 5
Preface

The product

SAP NetWeaver Identity Management Identity Center is a high-end identity management solution, capable of handling a large amount of repositories containing an unlimited amount of information. The Identity Center offers a robust, flexible and scalable high-availability solution for workflow, provisioning, data synchronization and joining for a large number of data repositories. The Identity Center provides a framework for a number of jobs created by the Data Synchronization Engine.

The reader

This manual is written for people who are to implement synchronization of Identity Center configurations between installations (staging).

Prerequisites

To get the most benefit from this manual, you should have the following knowledge:

- Thorough understanding of the Identity Center.

The manual

The manual contains a tutorial consisting of sections describing the issues that need to be considered when implementing an Identity Center configuration that should be prepared for a staging environment.

This tutorial is not a substitution for training.

Person names used in this tutorial are fictional.

Related documents

You can find useful information in the following documents:

- SAP NetWeaver Identity Management Identity Center tutorials and help file

© Copyright 2009 SAP AG. All rights reserved.
# Table of contents

**Introduction**

Section overview .................................................................................................................. 1

**Section 1: Configuration considerations**

GUIDs ........................................................................................................................................................2
Identity store objects .................................................................................................................... 3
Summary ............................................................................................................................................ 9

**Section 2: Performing a synchronization**

Exporting the configuration ........................................................................................................ 10
Bootstrapping the staging environment..................................................................................... 10
Performing an update on the production environment ........................................................... 11
Handling removed tasks .............................................................................................................. 11
Preparing a silent staging ............................................................................................................ 13

© Copyright 2009 SAP AG. All rights reserved.
Introduction

The Identity Center is often used in business critical operations. To be able to test and add new functionality, as well as software upgrades, it is important to have a test environment, which is identical to, but separate from the production environment. In the test environment new functionality and configurations can be verified, without the danger of disrupting the ongoing production. When all updates are verified in the test environment, the configuration updates are synchronized from the test environment to the production environment. This separation of test and production environments is referred to as staging.

Configuration Synchronization

When you create a configuration with the Identity Center that should be used in such an environment, there are some issues that should be considered and implemented in a specific way for this to be done as smoothly as possible.

This document describes the issues you need to consider and the necessary steps to perform bootstrapping and updates to the production environment.

Section overview

The document consists of the following sections:

<table>
<thead>
<tr>
<th>Section 1: Configuration considerations</th>
<th>This section gives an overview of issues that should be considered when designing a configuration that should be implemented in a staging environment.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 2: Performing a synchronization</td>
<td>Here you find a task list that must be done on the test and production environments to perform staging.</td>
</tr>
</tbody>
</table>
Section 1: Configuration considerations

This section describes issues that must be considered when building a configuration for a test/production scenario. This is also called a staging environment. Any new features or versions are introduced in the test environment before they are transferred to the production environment.

The synchronization utility is used for synchronizing configuration updates. To perform synchronization, the configuration is exported from the test environment. The resulting file(s) must be transferred to the production environment and imported there.

An import or update is done within a transaction. If something fails, the transaction is rolled back, and nothing is changed.

GUIDs

The key objects within the Identity Center configuration are identified using a GUID (Global Unique Identifier). This identifier is generated when the object is created in the Identity Center database, and guarantees uniqueness of the object across systems.

When importing the configuration, the import process checks if the same GUID exists in the production environment as in the import file. The following can occur:

- The GUID does not exist in the production environment. In this case, the user will only have the option to import the configuration. All GUIDs will be kept.
- The GUID exists in the production environment. In this case, the user is given two options:
  - Update. In this case, the existing object will be updated with changes from the imported file.
  - Import. In this case, the contents of the import file will be imported as new objects, and new GUIDs are generated for the objects.

Note:
It is not possible to perform an update of a configuration which is imported in this way.
Identity store objects

The main challenge for proper handling of configuration updates are references between objects in the configuration. To be able to handle this properly, there are several issues you need to be aware of. These are discussed in the following sections.

Global constants

When exporting any object, all the global constants will always be exported. The reason for exporting all global constants is that it is complicated to detect which global constants are actually in use. This would require parsing of all jobs, and all scripts within these jobs. As the constants can also be referenced indirectly, there is a risk that not all usage is detected.

In the staging scenario this should not cause any problems.

When importing or updating a configuration, existing global constants are never overwritten, only created if they do not exist. The reasoning behind this is that the global constants will often reference local resources, which are different in the test and production environments. One example is a reference to an SMTP server for sending e-mail.

After bootstrapping the configuration, it is important to consider changes to global constants in the test environment, and if these changes are relevant to the production environment, they must be implemented here manually.

Global variables and job variables

Neither global variables nor job variables are exported. They are only referenced from scripts, and the script itself must take care of this.

Repositories

If the export contains one or more jobs or tasks which references repositories, the corresponding repositories are also exported, including all constants and variables. This is also true for repositories which are referenced from attributes (owner repository or external attribute reference) and from the identity store (User Interface repository).

When importing or updating a configuration, you can select if you want to update (merge) repository constants or not.

If you leave "Update repository constant(s)" empty, existing repositories definitions are never overwritten or modified, but only created if they do not exist. Even if a new constant or variable is added to the repository in the test environment, this is not added on the production environment.

If you select "Update repository constant(s)", new repository constants are added to the repository definition in the production environment. Repository constants are never overwritten.
Identity store references

The identity store is referenced from all "To Identity store"-passes, as well as To-passes which use the identity store as source. If an identity store is selected from the drop down, this is stored as a numeric value, but will be converted to the GUID of the identity store when exporting. If an identity store with the given GUID exists when importing/updating, this will be re-mapped. If it does not exist, it will be set to "--none--", and will have to be set manually.

As an alternative you can use a global constant to reference the ID of the identity store. You can then use this global constant in the drop-downs for the identity store. (Note that it is not possible to use the context menu to insert values in a drop-down text box).

The definition of the global constant would be like this:

```
Global constant

Name: IDS_ID
Value: 5
Description: Encrypted value
```

The global constant is used in the following way in a To-pass:

```
To ASCII file

Repository Source    Destination    Data    Documentation
Identity store        Use identity store
SQL statement: SELECT DISTINCT maskkey FROM mask_entries WHERE rs_id=5
```

After the initial import on the production environment, just make sure to update the global constant correctly, as the global constants will never be changed by import or update.

Finally it is possible to use the reference "-- Self --" in passes which runs as action tasks. This indicates to use the identity store where the task is executing.
References to identity store entries

Always use the MSKEYVALUE when referencing entries (for example roles and privileges) in the identity store, and not MSKEY. The latter is guaranteed to be different on the production environment.

Note: Make sure that all referenced MSKEYVALUEs are available in the identity store of the production environment before you import.

Attributes

When exporting an identity store, the attribute definitions are also exported. Exporting any other object within the identity store will not export the attribute definitions, so it is important that these are in place before exporting and importing other objects which depends on the attribute definitions, such as tasks.

When importing an identity store, non-existent attributes are created. You will have an option whether to keep or overwrite attribute definitions. Internal attributes (i.e. all attributes starting with MX) are never overwritten.

Attribute restrictions

If any of the validation parameters on the attribute is set to a more restrictive value, this update is still performed, even though it may violate the data in the identity store on the production environment. Such updates must be done with care, as this can cause problems in the future if tasks or jobs assume that the data comply with the stricter rules.

This includes the following settings:

- Setting the "Mixed case characters" option
- Setting the "Mixed letters and number" options
- Setting the "Unique value" options
- Increasing the minimum length
- Decreasing the maximum length

In addition to these validation parameters, removing the "Multivalue attribute" will also be handled in the same way as described above.

Event tasks

When importing attributes having event tasks, these will be reconnected on import. This will not be a problem when exporting and importing an entire identity store, as the tasks will always be present.

However, if exporting only the schema information, event tasks will not be part of the export, and a warning is issued, and the event task is set to "-- None --".

Attribute values query

If an attribute uses the SQL query for defining the legal attribute values, the synchronization utility is unable to perform any sort of validating of this, and it may reference tables or views which do not exist on the production environment. It is therefore important to ensure that the given SQL statement is valid on the production environment, if the attribute is being updated.
Section 1: Configuration considerations

SAP NetWeaver Identity Management Identity Center Implementation guide - Staging environment

Scripts
If the attribute uses client or server side scripts for validation, this script must be manually added on the production environment.

Entry types
When exporting an identity store, the entry type definitions are also exported. Exporting any other object within the identity store will not export the entry type definitions, so it is important that these are in place before exporting and importing other objects which depends on the entry type definitions, such as tasks.

When importing an identity store, non-existent entry types are created. Modified entry types are updated in the production environment, with the following exceptions:

Entry type restrictions
If the change to the entry type may cause problems with data in the identity store, the update is not performed, and a warning is given. This includes the following:

- If attributes have been removed in the test environment, these are not removed in the production environment, as this may result in values being removed from the identity store, as they are no longer allowed on the entry type.
- If an attribute is set to mandatory, which was not mandatory before, this is not set to mandatory, as there might be entries within the identity store which does not have this attribute, and this would violate the new setting.

Event tasks
When importing entry types having event tasks, these will be reconnected on import. This will not be a problem when exporting and importing an entire identity store, as the tasks will always be present.

However, if exporting only the schema information, event tasks will not be part of the export, and a warning is issued, and the event task is set to "-- None --".
Section 1: Configuration considerations
SAP NetWeaver Identity Management Identity Center Implementation guide - Staging environment

Tasks
When a task is exported, the entire task hierarchy below that task is also exported, including all tasks and jobs, and the referenced repositories. If any task in the hierarchy references a task outside this hierarchy warnings are given, and the reference will be removed. This includes the following references:

- Execute task on initialize
- Task results (Execute on OK/Fail and Chain OK/Fail)

The attribute list for the task is not exported, unless the entire identity store is exported. This means that normally, all the referenced attributes must exist in the production environment to allow import of the task.

When importing you can choose an option that allows for importing a task which references non-existing attributes, but this means that all the missing attributes are removed from the task when importing.

If the referenced entry type does not exist in the production environment, this will be set to "-- None --".

Any access control involving a filter may cause problems. See below.

Other access controls may reference entries in the identity store, and these entries are not handled by the synchronization utility, which means that the entries with the same name must exist in the production environment's identity store, or the access control entry will be lost when importing.

For information about removing tasks from a task structure, see page 11.

Filters
Any use of filters may cause problems. The filter can be used as the access control in tasks, and as source in a To-pass.

If the filter is created using the wizard it should normally work OK, but if any modifications are made to the filter, there is a danger that it will not work properly on the production environment.

The export process will try to locate any identity store reference within the filter, i.e. "is_id" and "ocisid" as part of the filter. The value following the = will be replaced with the GUID of the given identity store upon export, and reconnected with the correct identity store (if it exists) when updating the production environment.

```
SELECT DISTINCT mskey FROM mxiv_entries WHERE is_id=1 AND 
((mskey IN (SELECT mskey FROM mxiv_entries WHERE attrname='LOCATION' AND
searchvalue = 'TRONDHEIM'))
AND (mskey IN (SELECT mskey FROM mxiv_entries WHERE attrname='TITLE' AND
searchvalue = 'MANAGER')))
```

If custom specific filters are required, it is possible to use global constants as part of the query.

Audit flags
All audit flags referenced by tasks will be exported.

When importing, audit flags will not be overwritten, but only created if they do not exist.
Job scripts

Local jobs scripts will always be overwritten on the production environment.

The synchronization utility does not perform any logic on the scripts. If script needs references to for example tasks or other identifiers which may be changed during synchronization, global constants should be used for this.

Global scripts

The user can decide whether to overwrite the global scripts or not.

As for local scripts, the synchronization utility does not perform any logic on the script. Internal references should be done using global constants.

Event agents

Event agent definitions are not handled by the synchronization utility, and must be set manually on the production environment after the first import.

When updating the production environment, the event agent definitions are not changed.

Dispatchers

Before performing the first import, the dispatcher(s) should be created on the production environment, using the same name as the test environment. The synchronization utility does not handle the dispatcher definitions, but will link any jobs imported on the production environment to the dispatcher(s) with the same name as on the test environment.

If no matching dispatcher(s) are found, it will use the dispatcher which is defined as default.

Job definitions

When importing or updating, it is possible to define whether the imported or updated jobs should be automatically enabled, and automatically added to revision control.
## Summary

The following is a list you should be aware of during the update process.

<table>
<thead>
<tr>
<th>Category</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global constants</td>
<td>Will only be created, never overwritten on the production environment.</td>
</tr>
<tr>
<td>Global variables</td>
<td>Are not exported or imported.</td>
</tr>
<tr>
<td>Job variables</td>
<td>Are not exported or imported.</td>
</tr>
<tr>
<td>Repositories</td>
<td>Will only be created, never overwritten on the production environment.</td>
</tr>
<tr>
<td>Identity store</td>
<td>In some cases, identity store references may not be handled properly.</td>
</tr>
<tr>
<td>Attributes</td>
<td>If changes to the attributes will put more restrictions on entries in the identity store, these updates will not be done, and a warning will be given on import. Attribute values query and scripts may not be handled properly.</td>
</tr>
<tr>
<td>Entry types</td>
<td>If changes to the entry type will put more restrictions on entries in the identity store, these updates will not be done, and a warning will be given on update. Be careful when using filters in access control, as they may not be handled properly.</td>
</tr>
<tr>
<td>Tasks</td>
<td>Conditional tasks, see filters.</td>
</tr>
<tr>
<td>Filters</td>
<td>In general, filters may cause problems during synchronization.</td>
</tr>
<tr>
<td>Audit flags</td>
<td>Audit flags will only be created, never overwritten.</td>
</tr>
<tr>
<td>Job scripts</td>
<td>References to task numbers, job number or other local objects.</td>
</tr>
<tr>
<td>Global scripts</td>
<td>Same as job scripts.</td>
</tr>
<tr>
<td>Event agents</td>
<td>Are not handled.</td>
</tr>
<tr>
<td>Dispatchers</td>
<td>Be sure to create dispatchers with the same name on the production environment.</td>
</tr>
</tbody>
</table>
Section 2: Performing a synchronization

This section describes how to synchronize configuration between the test and the production environments. It is assumed that you are familiar with the procedures involved in performing the necessary operations. For details, see the relevant tutorials or help file for the product.

Note:
Make sure to use the same version of the Import/Export tool when performing the synchronization between the test and production environments.

Exporting the configuration

The following steps must be performed on the test environment in the sequence they are listed here:

- Export job groups
- Export identity stores

Bootstrapping the staging environment

The following steps must be performed to prepare for staging on the production environment:

- Create dispatchers with the same names as on the test environment.
- Create event agents.
- Define the default import settings: dispatcher, enable jobs and check-in jobs.
- Create roles, privileges and dynamic groups (if used for task access control).
- Import job group(s).
- Import identity store(s).
- Manually link any event agent references.
- Manually update global constants and repositories.
- Manually update provisioning/deprovisioning task references on privileges.

Note:
The bootstrapping can also be done by creating a test environment from an existing production environment. Later updates can then be done from the test environment to the production environment.

Note:
After the bootstrapping, the configuration on the production environment (with the exception of constants, repositories, event agents and dispatchers) should not be changed, as this may cause problems when performing updates at a later time.
Performing an update on the production environment

When updates are to be synchronized from the test to the production environment, do the following:

1. In the test environment: Export job groups and identity stores as described on page 10.
2. In the test environment: Verify new roles, privileges and dynamic groups are introduced on the test environment which needs to be present.
3. In the production environment: Create the items identified in the previous step before performing the update on the production environment.
4. In the production environment: Stop any running dispatchers and wait for all jobs to enter idle state.
5. In the production environment: Updated (import) job group(s).
6. In the production environment: Update (import) identity store(s).
7. Start the dispatchers.

Handling removed tasks

When tasks are removed from the configuration in the test system, this must be handled specially when updating the production environment. If the provisioning queue for the task to be removed is empty, the task is removed from the production system.

If there is a queue of entries waiting to be processed, the task will be detached from its position in the task tree and moved to a folder called "MX:Pending-Delete". This folder will not be visible in the User Interface.

All references to the task will be removed, except from the following cases:

- Task references to this task from entries in the identity store, for instance MX_ADD_MEMBER.
- Task references from repositories and global constants, for instance MX_MODIFY.

Note: Make sure to change such references manually in the production environment if necessary.

All entries in the provisioning queue will be processed, but no new entries will be added to the queue.

The task will be removed with the next update of the production environment.

Task dependencies

Special care must be taken when if there is any kind of dependency between tasks in a task tree. If you remove a task that another task depends on, this task may not be able to process the entries in its queue when the other task is removed.

In this case you should rather make a copy of the entire task tree remove the previous version.
Some examples

**Removing a top task**

In this example, the task is deleted from the test environment. When importing the configuration, it will not be possible to delete the task from the production system, since there are entries in the queue.

It will be moved to the "MX:Pending-Delete" folder while processing the entries in the queue.

The task will not be available from the User Interface, to prevent new entries being added to the queue. The task will also be unlinked, if the task is also linked to another task.

Child tasks are *not* unlinked in this case, as the whole task including sub-tasks should be able to complete.

**Removing a sub-task**

In this case, one of the sub-tasks is deleted. When importing, it will only be necessary to unlink the task from all parent tasks.

The task will be moved to the "MX:Pending-Delete" folder.

The task will continue to process the queue until empty.

At the next import, it will also be detected that the task has been deleted, and if the queue is empty at this time, the task will physically be deleted.
Preparing a silent staging

The synchronization utility (export or import) is normally started from the Management console. This is a manual process that requires that someone runs through the wizard. It is also possible to create a batch job, and running the synchronization utility without the user interface.

Preparing a silent export

Start with performing a manual export of the data you want to export. You do not have to perform an export, but exit the synchronization user interface when it is started. Then make a copy of the file `export_start.bat`, which is located in the installation directory. You then need to add the keyword `SILENT` to the copy of the file, as shown below:

```bash
if %OPERATION%==M goto metadata
"C:\Program Files\SAP\IdM\sapjvm_5\bin\java.exe" -DSE_HOME="C:\Program Files\SAP\IdM\Identity Center" ccom.sap.idm.ic.syncutil.Export %DSEJ_HANDLE% SILENT %OPERATION% %START_ID% %JDBC_DRIVER% %JDBC_URL% "%IMXPORT_LOGFILE%" "C:\Program Files\SAP\IdM\Identity Center\Jobs\Test_Identity store_2.mcc"
goto cleanup
```

You may also want to change the parameter `IMXPORT_LOGFILE` in the file, which is where the log is stored.

```bash
SET IMXPORT_LOGFILE=C:\Program Files\SAP\IdM\Identity Center\ImXport.log
```

You may also want to change the target filename, which is the last parameter.

You can export the data by running this batch file.

Preparing a silent import

Start with performing a manual import. Then create a copy of the file `import_start.bat`. You then add the keyword `SILENT=I`.

```bash
REM --- Start import
"C:\Program Files\SAP\IdM\sapjvm_5\bin\java.exe" -Xms128m -Xmx512m -DSE_HOME="C:\Program Files\SAP\IdM\Identity Center" com.sap.idm.ic.syncutil.Import %DSEJ_HANDLE% SILENT=I OPERATION=%OPERATION% STARTID=%START_ID% JDBC_DRIVER=%JDBC_DRIVER% JDBC_URL=%JDBC_URL% "LOGFILE=%LOGFILE%" "INIFILE=%INIFILE%"
```

You may want to change the logfile and inifile parameters:

```bash
SET INIFILE=C:\Program Files\SAP\IdM\Identity Center\Jobs\JES IdS_Identity store_1.mcc
SET LOGFILE=C:\Program Files\SAP\IdM\Identity Center\ImXport.log
```

The parameter after `SILENT=`, can have one of the following values:

I: Perform an import. This is what you use the first time, to import the initial data.

O: Overwrite existing data. This is the normal operation when performing an update.

U: The existing values will be used, i.e. only new objects are imported. Normally not used.

There are some options that can be added to the command line:

<table>
<thead>
<tr>
<th>Option</th>
<th>Possible values</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMPORT_REPOSITORY_CONSTANTS</td>
<td>Specifies how repository constants are handled:</td>
</tr>
<tr>
<td></td>
<td>1: Import new repository constants</td>
</tr>
<tr>
<td></td>
<td>0: Never add new constants to existing repositories (default)</td>
</tr>
</tbody>
</table>
### LEAVE_GROUPS_AND_TASKS
Specifies whether groups, tasks and jobs that no longer exist in the source (for instance, have been removed) should be deleted from the target system.

- **1**: Leave groups, tasks and jobs that do not exist in the source
- **0**: Delete groups, tasks and jobs that do not exist in the source (default)

### UPDATE_EVENT_TRIGGERS
Specifies whether event triggers for attributes and entry types should be updated when importing provision groups and tasks.

- **1**: Update event triggers for attributes and entry types
- **0**: Do not update event triggers for attributes and entry types (default)