HP OpenView Select Identity

Connector for IBM Tivoli Access Manager

Installation and Configuration Guide

Connector Version: 3.4
Select Identity Version: 3.3.1

August 2005

Legal Notices

Warranty

Hewlett-Packard makes no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

A copy of the specific warranty terms applicable to your Hewlett-Packard product can be obtained from your local Sales and Service Office.

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company
United States of America

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).

Copyright Notices

This product includes software developed by the Apache Software Foundation (http://www.apache.org/).

Select Identity uses software from the Apache Jakarta Project including:

- Commons-beanutils.
- Commons-collections.
- Commons-logging.
- Commons-digester.
- Commons-httpclient.

- Element Construction Set (ecs).
- Jakarta-poi.
- Jakarta-regexp.
- Logging Services (log4j).

Additional third party software used by Select Identity includes:

- JasperReports developed by SourceForge.
- iText (for JasperReports) developed by SourceForge.
- BeanShell.
- Xalan from the Apache XML Project.
- Xerces from the Apache XML Project.
- Java API for XML Processing from the Apache XML Project.
- SOAP developed by the Apache Software Foundation.
- JavaMail from SUN Reference Implementation.
- Java Secure Socket Extension (JSSE) from SUN Reference Implementation.
- Java Cryptography Extension (JCE) from SUN Reference Implementation.
- JavaBeans Activation Framework (JAF) from SUN Reference Implementation.
- OpenSPML Toolkit from OpenSPML.org.
- JGraph developed by JGraph.
- Hibernate from Hibernate.org.
- BouncyCastle engine for keystore management, bouncycastle.org.

Trademark Notices

HP OpenView Select Identity is a trademark of Hewlett-Packard Development Company, L.P.

Microsoft, Windows, the Windows logo, and SQL Server are trademarks or registered trademarks of Microsoft Corporation.

Sun™ workstation, Solaris Operating Environment™ software, SPARCstation™ 20 system, Java technology, and Sun RPC are registered trademarks or trademarks of Sun Microsystems, Inc. JavaScript is a trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

This product includes the Sun Java Runtime. This product includes code licensed from RSA Security, Inc. Some portions licensed from IBM are available at http://oss.software.ibm.com/icu4j/.

IBM, DB2 Universal Database, DB2, WebSphere, and the IBM logo are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group.

This product includes software provided by the World Wide Web Consortium. This software includes xml-apis. Copyright © 1994-2000 World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. http://www.w3.org/Consortium/Legal/

Intel and Pentium are trademarks or registered trademarks of Intel Corporation in the United States, other countries, or both.

AMD and the AMD logo are trademarks of Advanced Micro Devices, Inc.

BEA and WebLogic are registered trademarks of BEA Systems, Inc.

VeriSign is a registered trademark of VeriSign, Inc. Copyright © 2001 VeriSign, Inc. All rights reserved.

All other product names are the property of their respective trademark or service mark holders and are hereby acknowledged.

Support

Please visit the HP OpenView web site at:

http://www.managementsoftware.hp.com/

This web site provides contact information and details about the products, services, and support that HP OpenView offers.

You can also go directly to the support web site at:

http://support.openview.hp.com/

HP OpenView online software support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valuable support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit and track progress on support cases
- Manage a support contract
- Look up HP support contacts
- Review information about available services
- Enter discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and log in. Many also require a support contract.

To find more information about access levels, go to:

http://support.openview.hp.com/access_level.jsp

To register for an HP Passport ID, go to:

https://passport2.hp.com/hpp/newuser.do

contents

1

Installing the Connector

The IBM Tivoli Access Manager connector — hereafter referred to as the TAM connector — enables HP OpenView Select Identity to perform the following tasks in Tivoli Access Manager (TAM):

- Add, update, and remove users
- Retrieve user attributes
- Enable and disable users
- Verify a user's existence
- Change user passwords
- Reset user passwords
- Expire passwords
- Validate passwords
- Retrieve all entitlements
- Retrieve a list of supported user attributes
- Assign and unassign entitlements to and from users

It is a one-way connector and pushes user data from the Select Identity database to the target Tivoli Access Manager. The TAM connector uses Java APIs to communicate with Tivoli Access Manager. The mapping file controls how Select Identity fields are mapped to Tivoli Access Manager fields.

The TAM connector is packaged in the following files:

- `TamClient.jar` — contains the TAM client Java classes, which implement the TAM APIs to access and manage the TAM resource
- `TamSchema.jar` — contains the `tivoliaccessmanager.xml` file, which maps attributes from Select Identity to TAM
- `TamConnector.rar` — contains the RAR file, which contains the TAM connector implementation -elated Java class files, third party JAR files, and the TAM property file
- `tam-scripts.tar.gz` — contains the `tamcfg.ksh` script for Solaris systems, which is used to configure the PD JRE, create the configuration and keystore files, and to invoke the TAM client APIs
- `tam-scripts.zip` — contains the `tamcfg.bat` script for Windows-based systems, which is used to configure PD JRE, create the configuration and keystore files, and invoke the TAM client APIs

These files are located in the `IBM Tivoli Access Manager` directory on the Select Identity Connector CD.

System Requirements

The TAM connector is supported in the following environment:

Select Identity Version	Application Server	Database	TAM Version and Operating System
3.0.2	WebLogic 8.1.2 on Solaris 9	Oracle 9i	4.1 on Solaris 8 5.1 on Solaris 9
	WebLogic 8.1.2 on Windows 2003	SQL Server 2000	5.1 on Windows 2000
	WebSphere 5.1.1 on Windows 2000	DB2 8.2 (or DB2 8.1 Service Pack 7)	5.1 on Windows 2000 and Solaris 9
	WebSphere 5.1.1 on Solaris 9	DB2 8.2 (or DB2 8.1 Service Pack 7)	5.1 on Windows 2000

Select Identity Version	Application Server	Database	TAM Version and Operating System
3.3	WebLogic 8.1.4 on Solaris 9	Oracle 9i	4.1 on Solaris 8
3.3.1	WebLogic 8.1.4 on Windows 2003	SQL Server 2000	5.1 on Solaris 9
	WebSphere 5.1.1 on HP-UX 11i	Oracle 9i	

This connector is supported with TAM 4.1 on Solaris 8 and TAM 5.1 on Windows 2000 and Solaris 9. Also, TAM is supported with the following:

- iPlanet as Directory Server on Windows 2000 and Solaris 9
- Tivoli Policy Server on Windows 2000 and Solaris 9
- Tivoli Authorization Server on Windows 2000 and Solaris 9

Also, the Tivoli Access Manager Runtime must be installed and configured before you install the TAM connector. The Installation Guide for Tivoli Access Manager is available at the following URL:

http://publib.boulder.ibm.com/tividd/td/ITAME/SC32-1362-00/en_US/HTML/am51_install.htm

Configuring TAM Java Runtime Environment

The TAM Java Runtime Environment (JRE) component enables Java applications to manage and use TAM security. Before deploying the connector, you must configure the TAM JRE. This enables the connector to access and provision users in TAM. This section explains the `tamcfg.bat` (for Windows) and `tamcfg.ksh` (for Solaris) scripts, which can be used to configure the TAM JRE.

1. Create a subdirectory in the Select Identity home directory where the TAM client will reside. For example, you could create the `C:\Select_Identity\tamclient` folder on Windows, or you could create the `/opt/Select_Identity/tamclient` directory on Solaris.

This TAM client directory will also store the `CFG.properties` and `KeyStore` that will be created using the `tamcfg` script.

2. On Windows, extract `tamcfg.bat` from the `tam-scripts.zip` file to the TAM client subdirectory. On UNIX, extract `tamcfg.ksh` from the `tam-scripts.tar.gz` file to the TAM client subdirectory.
3. Copy the `TamClient.jar` file from the Select Identity Connector CD to the TAM client subdirectory.
4. Make sure that all of the directories and files that are used to define the variables in `tamcfg.bat` or `tamcfg.ksh` exist with the required permissions. All of the variables are explained below.

   — JREHOME
   The JRE home directory. This must be the path to the IBM JDK JRE. Examples:

   JREHOME=/opt/WebSphere/AppServer/java/jre

   JREHOME=C:\Program Files\WebSphere\AppServer\java\jre

   Make sure `PolicyDirector` resides here and this folder contains the `java/export/pdjrte` folder with all of the TAM JAR files. If not, create these folders and simply copy the TAM JAR files here, which come with the TAM installation. Here is a listing of the files:

   `ibmjcefw.jar`

   `ibmjceprovider.jar`

   `ibmjsse.jar`

   `ibmpkcs.jar`

   `jaas.jar`

   `local_policy.jar`

   `PD.jar`

   — PDHOME
   The home directory of Tivoli Access Manager Policy Director runtime. Examples:

   PDHOME=$JREHOME/PolicyDirector

   PDHOME=%JREHOME%\PolicyDirector

   — PD_LIB_DIR
   The folder where Policy Server JAR files are located. Examples:

PD_LIB_DIR=$PDHOME/java/export/pdjrte

PD_LIB_DIR=%PDHOME%\java\export\pdjrte

— TAM_CLIENT_DIR
The folder where Select Identity's `TamClient.jar` is installed. Examples:

TAM_CLIENT_DIR=/opt/Select_Identity/tamclient

TAM_CLIENT_DIR=C:\Select_Identity\tamclient

This folder will also contain the TAM key store and configuration files. These files are generated by the `tamcfg` script and are referenced later.

— APP_SERVER_IP
The IP Address of the machine on which Select Identity will be running. Example:

APP_SERVER_IP=16.73.17.88

— POLICY_SERVER_IP
The IP Address of Tivoli Access Manager Policy Server. Example:

POLICY_SERVER_IP=15.70.184.141

— APP_SERVER_NAME
The name of the Select Identity application, which is used to create an account for the Select Identity application to access TAM. It is also used to create a registry user in TAM Policy Server. Example:

APP_SERVER_NAME=SI88aTam141

— PD_ADMIN_ID
The name of an administrative account created in Tivoli Access Manager. Select Identity uses this account for user provisioning. Example:

PD_ADMIN_ID=sec_master

— PD_ADMIN_PASSWD
The password for the administrative account (PD_ADMIN_ID).

— AUTH_SERVER_IP
The IP Address of Tivoli Access Manager Authentication Server. Usually this is the same as the machine on which the Policy Server is running. Example:

AUTH_SERVER_IP=$POLICY_SERVER_IP

— APP_MODE
Set to **remote** if the Select Identity application will run on a machine remote from the machine running the Tivoli Access Manager Policy Server. Example:

APP_MODE=remote

— OPERATION
The operation to be performed with SvrSslCfg. For the first creation of the key store and configuration file, this must be set to **create**. If there is any changes to the other variables, specify **replace** for regeneration.

OPERATION=create

5 After verifying for the existence of all files and directories, run the following command to configure the PD JRE component. Pass **jrccfg** as the argument to the script.

On Solaris:
tamcfg.ksh jrtcfg

On Windows:
tamcfg.bat jrtcfg

If an "Authentication method is unavailable" error occurs while running the `tamcfg` script, verify whether the Directory Server, Policy Server, and Authentication Server are running.

Creating the Property and Key Store Files

The connector uses secure communication with the TAM Policy Server. You must perform steps to generate the configuration property files and key store file.

The same script used in Configuring TAM Java Runtime Environment on page 9 (`tamcfg.bat` or `tamcfg.ksh`) can be used to create the configuration property file and key store file. First, you must configure the PD JRE then you can create the configuration and key store files.

The files will be created in the directory specified by the TAM_CONFIG_DIR variable. This is the same directory where you extracted the `tamcfg` script.

Complete the following steps to created the files:

1. Make sure that all of the directories and files that are used to define the variables in `tamcfg.bat` or `tamcfg.ksh` exist with the required permissions. See Step 4 on page 10 for an explanation of the variables.

   Also, note that this script uses the `com.tivoli.pd.jcfg.SvrSslCfg` Java class to create the required property and key store files.

2. Execute the `tamcfg` script as shown below :

   On Windows :
   `tamcfg.bat sslcfg`

   On Solaris:
   `tamcfg.ksh sslcfg`

   This command creates two files:

   APP_SERVER_NAME`_TAM_CFG.properties`
   APP_SERVER_NAME`_TAM_KEY.ks`

   where *APP_SERVER_NAME* is the name of the SI application that you specified in the `tamcfg` script. These files are used by the TAM connector client and should not be edited, moved, or deleted from this directory.

Verifying the Tivoli Access Manager Client

The Tivoli Access Manager client is a client application that uses TAM APIs to access and provision on the TAM resource. It takes command line arguments that support user operations such as listing users and groups, creating users and groups, and so on. You can use the TAM client to verify the connectivity to the TAM resource and to verify provisioning.

To use the TAM client, you can run the `tamcfg` script, which was used to configure the TAM JRE and to create of the property and key store files. Run the `tamcfg.bat` script (for Windows) or `tamcfg.ksh` script (for Solaris) from the command line in a similar way it was used in the previous procedures but with a different set of arguments. Both scripts take the **tamclient** argument to run the Java class TamClient. This class implements all user-related operations.

The following are the arguments to be passed to the `tamcfg` script for executing the commands.

`tamcfg tamclient` *`function appName keyField`*

where:

- **`tamclient`** is a constant that is always specified
- ***`function`*** is the name to be executed, such as isUserExists, getGroups, and so on.
- ***`appName`*** is the application name that you provided in the `tamcfg` script.
- ***`keyField`*** is the key field of the user. This argument is optional but necessary for such functions as isUserExists.

The following are examples used for performing various operations to validate whether the TAM client is working properly.

doTest

Use the doTest function to test connectivity to the TAM resource. If this test fails, it means that some of the configuration done in earlier steps is wrong or you do not have access to TAM Policy Server. In this case, you must edit the `tamcfg` script and verify that all variables are correct. Then, you must regenerate the property and key store files. The following is an example of the command run on Solaris:

`tamcfg.ksh tamclient doTest SI88aTam141`

where **`SI88aTam141`** is the application name provided in the `tamcfg` script. If the command runs successfully, the command returns `OK`.

isUserExists

Use this function to verify for the existence of a user in TAM. The following is an example of the command run on Solaris:

`tamcfg.ksh tamclient isUserExists SI88aTam141 tamuser01`

If the user exists, the command returns `OK`. However, if the user does not exist, the command returns `ERROR`.

getGroups

Use this function to list all groups that exist on the TAM resource. The following is an example of the command run on Solaris:

tamcfg.ksh tamclient getGroups SI88aTam141

Here is the output:

```
handleResult(msgs): ENTER
handleResult(msgs): EXIT-0
groupkey:::hr-mgrs
groupkey:::qa-mgrs
groupkey:::pd-mgrs
groupkey:::SecurityGroup
groupkey:::ivmgrd-servers
groupkey:::iv-admin
groupkey:::secmgrd-servers
groupkey:::webseal-servers
groupkey:::webseal-mpa-servers
groupkey:::ivacld-servers
groupkey:::remote-acl-users
OK
```

findUser

Use this function to retrieve the details of an existing user on the TAM resource. The following is an example of this command run on Solaris:

tamcfg.ksh tamclient findUser SI88aTam141 tamuser01

The following is the output:

```
User id = tamuser01
Description = Tivoli Access Manager 5.1 First User
Account valid = true
Password valid = true
Policy Director user = true
```

```
Has single-signon capabilities = false

Registry name = cn=tamuser01first tamuser01last-B07242BA-BAE7-077B-
B511-F90D8A93B899,ou=People,dc=qa,dc=trulogica,dc=com

First name = tamuser01first

Last name = tamuser01last

Groups =

User policy =

User id = tamuser01

Account expiration date = null (not enforced)

Maximum failed logins allowed before account disabled = 0 (not
enforced)

Account disable time interval (in minutes) = 0 (not enforced)

Spaces allowed in password = false (not enforced)

Maximum password age (in seconds) = 0 (not enforced)

Maximum repeated characters allowed in password = 0 (not enforced)

Minimum number of alphabetic characters required in password = 0
(not enforced)

Minimum number of non-alphabetic characters required in password =
0 (not enforced)

Minimum password length = 0 (not enforced)

Account is accessible on the following days = Any day (not
enforced)

Account accessible starttime (in minutes after midnight on
accessible days) = 0

Account accessible endtime (in minutes after midnight on accessible
days) = 0

OK
```

Deploying on the Web Application Server

To install the TAM connector on the Select Identity server, complete these steps:

1. Create a subdirectory in the Select Identity home directory where the connector's RAR file will reside. For example, you could create the `C:\Select_Identity\connectors` folder on Windows. (A connector subdirectory may already exist.)
2. Copy the `TamConnector.rar` file from the Select Identity Connector CD to the connector subdirectory.
3. If deploying the connector on WebLogic, complete the following steps. If deploying on WebSphere, skip to Step 4 on page 18.
   a. Create a schema subdirectory in the Select Identity home directory where the connector's mapping file(s) will reside. For example, you could create the `C:\Select_Identity\schema` folder. (This subdirectory may already exist.)
   b. Extract the contents of the `TamSchema.jar` file (on the Select Identity Connector CD) to the schema subdirectory.
   c. Ensure that the CLASSPATH environment variable in the WebLogic server startup script references the schema subdirectory.
   d. Start the application server if it is not currently running.
   e. Log on to the WebLogic Server Console.
   f. Navigate to ***My_domain*** →**Deployments** →**Connector Modules**.
   g. Click **Deploy a New Connector Module**.
   h. Locate and select the `TamConnector.rar` file from the list. It is stored in the connector subdirectory.
   i. Click **Target Module**.
   j. Select the **My Server** (your server instance) check box.
   k. Click **Continue**. Review your settings.
   l. Keep all default settings and click **Deploy**. The Status of Last Action column should display Success.

4 If deploying the connector on WebSphere, complete the following steps:

a Stop the application server.

b Extract the contents of the `TamSchema.jar` file (on the Select Identity Connector CD) to the `WebSphere\AppServer\lib\ext` directory.

c Start the application server.

d Log on to the WebSphere Application Server Console.

e Navigate to **Resources →Resource Adapters**.

f Click **Install RAR**.

g In the Server path field, enter the path to the `TamConnector.rar` file.

h Click **Next**.

i In the Name field, enter a name for the connector.

j Click **OK**.

k Click the **Save** link (at the top of the page).

l On the Save to Master Configuraton dialog, click the **Save** button.

m Click **Resources →Resource Adapters**.

n Click the new connector.

o Click **J2C Connection Factories** in the Additional Properties table.

p Click **New**.

q In the Name field, enter the name of the factory for the connector. For the SQL connector, enter **`eis/TamConnector`**.

r Click **OK**.

s Click the **Save** link.

t On the Save to Master Configuraton dialog, click the **Save** button.

u Restart WebSphere.

5 Modify the mapping file, if necessary. See Understanding the Mapping File on page 19 for details.

After installing the connector, see Configuring the Connector on page 27 to register and configure the connector in Select Identity.

2

Understanding the Mapping File

The TAM connector is deployed with the `tivoliaccessmanager.xml` mapping file, which describes the attributes required by the system. The file is created in XML, according to SPML standards, and is bundled in a JAR file called `tamschema.jar`. The mapping file is used to map user account additions and modifications from Select Identity to the system resource. When you deploy a resource using the Resources page of the Select Identity client, you can review this file.

You can create attributes that are specific to Select Identity using the Attributes page in the Select Identity client. These attributes can be used to associate Select Identity user accounts with system resources by editing the connector mapping file described in this chapter. This process becomes necessary because, for example, a single attribute “owner” can have a different name on different resources, such as “OWNER” for Tandem, “UID” for a database, and “ownerID” on a Windows server.

This file does not need to be edited unless you want to map additional attributes to your resource. If attributes and values are not defined in this mapping file, they cannot be saved to the resource through Select Identity.

General Information

The following operations can be performed in the mapping file:

- Add a new attribute mapping
- Delete an existing attribute mapping
- Modify attribute mappings

Here is an explanation of the elements in the XML mapping file:

- **<Schema>**, **<providerID>**, and **<schemaID>**

  Provides standard elements for header information.

- **<objectClassDefinition>**

  Defines the actions that can be performed on the specified object as defined by that name attribute (in the <properties> element block) and the Select Identity-to-resource field mappings for the object (in the <memberAttributes> block). For example, the object class definition for users defines that users can be created, read, updated, deleted, reset, and expired in Tivoli Access Manager.

  - **<properties>**
    Defines the operations that are supported on the object. This can be used to control the operations that are performed through Select Identity. The following operations can be controlled:

    — Create (CREATE)

    — Read (READ)

    — Update (UPDATE)

    — Delete (DELETE)

    — Enable (ENABLE)

    — Disable (DISABLE)

    — Reset password (RESET_PASSWORD)

    — Expire password (EXPIRE_PASSWORD)

    — Change password (CHANGE_PASSWORD)

The operation is assigned as the name of the <attr> element and access to the operation is assigned to a corresponding <value> element. You can set the values as follows:

— true — the operation is supported by the connector

— false — the operation is not supported by the connector

— bypass — the operation is not supported by the connector

Here is an example:

```
<objectClassDefinition name="User" description="TAM User">
  <properties>
    <attr name="CREATE">
      <value>true</value>
    </attr>
    <attr name="READ">
      <value>true</value>
    </attr>
...
```

- **<memberAttributes>**
  Defines the attribute mappings. This element contains <attributeDefinitionReference> elements that describe the mapping for each attribute. Each <attributeDefinitionReference> must be followed by an <attributeDefinition> element that specifies details such as minimum length, maximum length, and so on.

  Each <attributeDefinitionReference> element contains the following attributes:

  — Name — the name of the reference.

  — Required— if this attribute is required in the provisioning (set to true or false).

  — Concero:tafield — the name of the Select Identity resource attribute. In general, the attribute assigned to tafield should be the same as the physical resource attribute, or at least the connector attribute. For example, it is recommended to have the following:

  ```
  <attributeDefinitionReference name="FirstName"
  required="false" concero:tafield="[givenname]"
  concero:resfield="givenname" concero:init="true"
  concero:isMulti="true"/>
  ```

instead of this:

```
<attributeDefinitionReference name="FirstName"
required="false" concero:tafield="[FirstName]"
concero:resfield="givenname" concero:init="true"
concero:isMulti="true"/>
```

— Concero:resfield — the name of the physical resource attribute from the resource schema. If the resource does not support an explicit schema (such as UNIX), this can be a tag field that indicates a resource attribute mapping.

  Also, the attribute name may be case-sensitive; for example, if the attribute is defined in all uppercase letters on the resource, be sure to specify it in all uppercase letters here.

— Concero:isKey — An optional attribute that, when set to true, specifies that this is the key field to identify the object on the resource. Only one <attributeDefinitionReference> can be specified where isKey="true". This key field does not need to be the same as the key field of the identity object in Select Identity.

  Note that for a key field mapping where isKey=”true” and tafield is not assigned the UserName attribute, UserName should not be used in any other mapping. That is, UserName can be assigned to tafield only in cases where it is mapped to the key field in the resource. Example:

```
<attributeDefinitionReference name="UserName"
required="true" concero:tafield="[UserName]"
concero:resfield="uid" concero:isKey="true"
concero:init="true"/>
```

— Concero:init — An optional attribute that identifies that the attribute is initialized with the value of the attribute passed in from Select Identity.

Here is an example:

```
<memberAttributes>
  <attributeDefinitionReference name="TAM-UName"
   required="true" concero:tafield="User Name"
   concero:resfield="username" concero:isKey="true" />
...
```

The interpretation of the mapping between the connector field (as specified by the Concero:tafield attribute) and the resource field (as specified by the Concero:resfield attribute) is determined by the connector. The TAM connector has code to interpret the mappings in one way, as follows:

- The connector attribute names are specified in square braces, like this: [xyz]. The value of attribute xyz is taken from the UserModel during provisioning.
- Composite attributes can be specified in the TAM connector mapping file. To do this, specify [attr1] xxxx [attr2] as the connector attribute. This specifies that the value of the attr1 and attr2 attributes should be combined with the string xxxx to form a mapping for the specified resource field. TAM connector has code to handle these composite mappings.

- **<attributeDefinition>**

  Defines the properties of each object's attribute. For example, the attribute definition for the Directory attribute defines that it must be between one and 50 characters in length and can contain the following letters, numbers, and characters: a-z, A-Z, 0-9, @, +, and a space.

  Here is an excerpt from the `tivoliaccessmanager.xml` file:

```
<attributeDefinition name="GroupName" description="GroupName"
type="xsd:string" >
  <properties>
    <attr name="minLength">
      <value>1</value>
    </attr>
    <attr name="maxLength">
      <value>8</value>
    </attr>
    <attr name="pattern">
      <value><![CDATA[[a-zA-Z0-9@]+]]> </value>
    </attr>
  </properties>
</attributeDefinition>
```

- **<concero:entitlementMappingDefinition>**

  Defines how entitlements are mapped to users.

- **<concero:objectStatus>**

  Defines how to assign status to a user.

- **<concero:relationshipDefinition>**

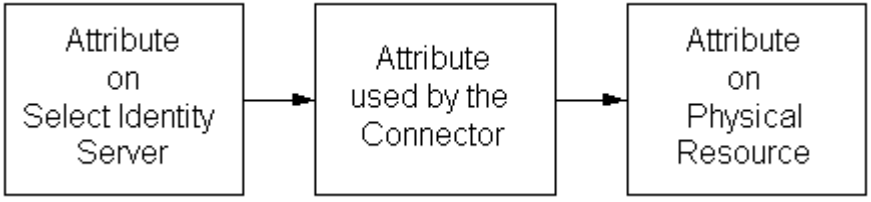
  Defines how to create relationships between users.

Tivoli Access Manager Mapping Information

The following is a description of the columns provided in the tables below:

- **Select Identity Resource Attribute —** The name of the attribute on the Select Identity server.
- **Connector Attribute**— The attribute used by the TAM connector.
- **Actual Attribute on Tandem Server** — The name of the attribute on the TAM server, to which the Select Identity and logical resource attributes are mapped. These attributes cannot be changed.
- **Description** — A description of the attribute and any noteworthy information needed when assigning values to the attribute.

Here is an illustration of how the attributes are related:



The attributes on the TAM server must be mapped to Select Identity attributes in the `tivoliaccessmanager.xml` file. For TAM attributes that cannot be mapped to existing Select Identity attributes, you must create new attributes in Select Identity and map them to the corresponding TAM attributes. Also, some connector attributes must be mapped in a different way than normal attributes in order to leverage advanced Select Identity functionality. See the Description column for more information.

The following table contains the attributes specific to iPlanet LDAP, which is used as the Directory Server for TAM.

Select Identity Resource Attribute	Attribute on Connector	TAM User Attribute	Attribute on Physical Resource (iPlanet)	Description
GUID	GUID	cn		The user's global ID.
[First Name] [Last Name]-[GUID]	cn	Part of Registry Name (DN)	cn	The user's common name.
User Name	uid	UserName	uid	A value from 1-100 alphanumeric characters in length.
Password*	Password	Password	userPassword	1-10 alphanumeric characters. This value is encrypted.
First Name	fname	First name	cn	A value from 1-50 alphanumeric (including '.') characters in length.
Last Name	lname	Last name	sn	A value from 1-50 alphanumeric (including '.') characters in length.
Description**	description	Description	description	A value from 1-100 alphanumeric characters in length.

* Select Identity does not allow a user to reset or change his or her password using the Self Service function. However, a Select Identity administrator can reset the password using the Reset Password functionality.

** Tivoli Access Manager allows the modification of the Description attribute only. You cannot edit any of the other attributes.

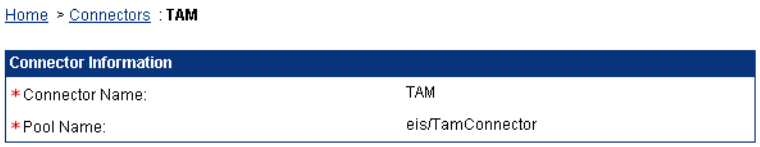
3

Configuring the Connector

After you deploy the connector on the application server, you must configure Select Identity to use the connector by deploying it in the Select Identity client. The following provides an overview of the procedures you must complete in order to deploy your connector. It also provides connector-specific information you must provide when configuring Select Identity to use the connector.

1 Register the connector with Select Identity by clicking the **Deploy New Connector** button on the Connectors home page. Complete this procedure as described in the "Connectors" chapter of the *HP OpenView Select Identity Administrator Guide*.

After you deploy the connector, the connector properties will look similar to this:

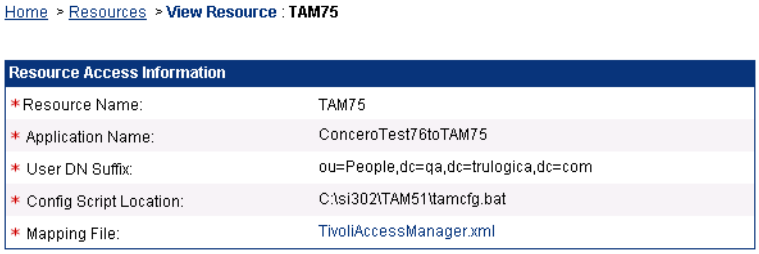


2 Deploy a resource that uses the newly created connector. On the Resources home page, click the **Deploy New Resource** button. When configuring the resource, refer to the following table for parameters specific to this connector:

Field Name	Sample Values	Description
Resource Name	TAM75	Name given to the resource.
Resource Type	TAM	The connector that was deployed in Step 1 on page 27.
Authoritative Source	No	Whether this resource is a system that is considered to be the authoritative source for user data in your environment. You must specify **No** because the connector cannot synchronize account data with the Select Identity server.
Associate to Group	Selected	Whether the system uses the concept of groups. For this connector, select this option.
Application Name	gvSI86TAM75	Name of the application configured to access TAM. This is the name given in the `tamcfg.bat` or `tamcfg.ksh` script while generating the configuration property and key store files.
User DN Suffix	ou=People,dc=qa,dc=HP	The complete DN suffix of the users in the Directory Store. This is where users will be provisioned.

Field Name	Sample Values	Description
Config Script Location	*On Windows*: C:\Select_Identity\tamclient\tamcfg.bat *On UNIX*: /opt/Select_Identity/tamclient/tamcfg.ksh	Full path to the location of the `tamcfg.bat` or `tamcfg.ksh` script, which is installed in the TAM client subdirectory.
Mapping File	TivoliAccessManager.xml	Name of the file that specifies the attribute mappings. This file should exist in the classpath of the application server. Click the **View** link to open the file in a browser window. If this file cannot be viewed, Select Identity could not locate the file.

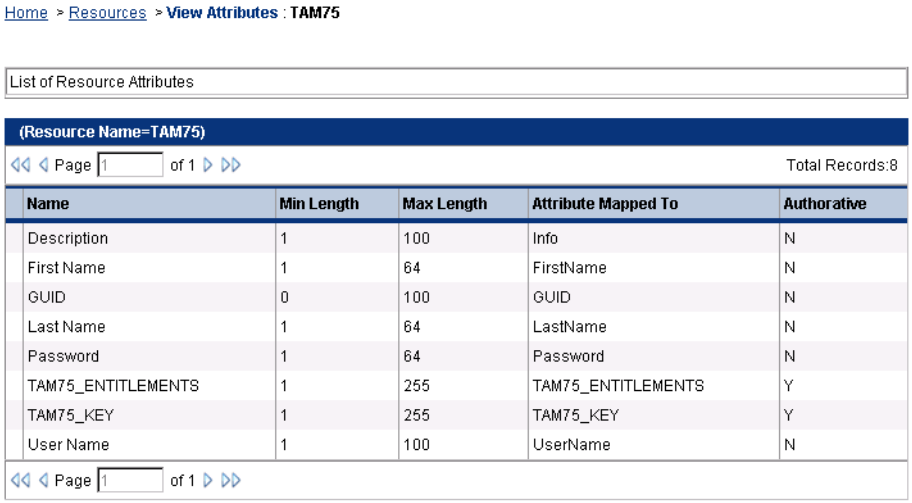
Complete the steps in this procedure as described in the "Resources" chapter of the *HP OpenView Select Identity Administrator Guide*. After you deploy the resource for the TAM connector, the Access Info page of the resource properties will look similar to this:



3 Create attributes that link Select Identity to the connector. For each mapping in the connector's mapping file, create an attribute using the Attributes capability on the Select Identity client.

Refer to the "Attributes" chapter in the *HP OpenView Select Identity Administrator Guide* for more information. After you create the attributes

for the TAM connector, the View Attributes page for the resource will look similar to this:



4 Create a Service that will use the newly created resource. To do so, click the **Deploy New Service** button on the Services home page. Complete this procedure as described in "Services" of the *HP OpenView Select Identity Administrator Guide*. You will reference your new resource created in Step 2 while creating this service.

4

Uninstalling the Connector

If you need to uninstall a connector from Select Identity, make sure that the following are performed:

- All resource dependencies are removed.
- The connector is deleted using the Select Identity client Connectors pages.

On WebLogic

Perform the following to delete a connector:

1. Log on to the WebSphere Application Server Console.
2. Navigate to **Resources** →**Resource Adapters**.
3. Select the TAM connector (adapter).
4. Click **Delete**.
5. Click the **Save** link (at the top of the page).
6. On the Save to Master Configuraton dialog, click the **Save** button.

On WebSphere

Complete the following steps to uninstall the connector on WebSphere:

1. Log on to the WebSphere Application Server Console.
2. Navigate to **Resources** →**Resource Adapters**.
3. Select the connector to uninstall.
4. Click **Delete**.
5. Click the **Save** link (at the top of the page).
6. On the Save to Master Configuraton dialog, click the **Save** button.

A

Troubleshooting

This appendix describes common problems seen during the installation and execution of the connector.

- While running the `tamcfg` script to generate the property or key store files, the following error may occur:

  ```
  Authentication method is unavailable
  ```

  Verify that the Directory Server, Policy, and Authentication servers are running.

- If creating a user, adding entitlements, or removing entitlements takes too long time or hangs, restart the Directory server.