

Swivel Core V4 Repository Menu

Contents

- 1 Introduction to Repositories
- 2 Servers
- 3 Types
- 4 Groups
- 5 Groups Rights
- 6 Attributes
- 7 XML
 - ◆ 7.1 A Note on Attributes in XML Repositories
- 8 Active Directory
- 9 ADAM
- 10 Simple LDAP
- 11 LDAP Writeable
- 12 Database
 - ◆ 12.1 A Note on Attributes

Introduction to Repositories

Users in the Swivel database can come from a number of sources. We refer to these as user **repositories**. Each user belongs to exactly one repository, and in order to import users from a repository, the repository details must be entered.

Currently, we support the following types of repository:

- XML - these are "built-in" repositories
- LDAP - including:
 - ◆ Active Directory - the most common form of LDAP repository
 - ◆ Simple LDAP - any LDAP repository can be configured using this
 - ◆ ADAM - now known as Active Directory Lightweight Directory Services, or ADLDS.
- External Database

Repositories can be read-only or writeable:

- Active Directory, Simple LDAP and Database are read-only repositories - Swivel cannot modify the users within the repository.
- XML, ADAM and LDAP Writeable are writeable repositories - Swivel can create, modify and delete users within these repositories.

Repositories import users into the Swivel database by running a **User Sync**. This can either be done manually, or at regular intervals, using a scheduled job. Writeable repositories will automatically sync changes made within the Swivel admin console to the Swivel database, but you will still need to run user sync if you make changes to those repositories outside the console.

Additionally, there are Agent repositories. Any Agent can be configured to act as a repository, in which case, users can be created, modified and deleted within the Swivel database using API calls. Agent repositories can be defined on top of other repository types, by giving them exactly the same name, but beware that running a User Sync on a repository will overwrite any changes made by the Agent.

The different repository types will be described in more detail later on, as we get to the settings menus.

Repositories can also be used to authenticate users' passwords. This is typically done for LDAP repositories, but XML and Database repositories are also supported.

Servers

Repository / Servers ?

Please add and configure the user repository servers.

Repository Servers:

▾ Local

Repository Name:

Local

Repository Type:

XML

Delete

▸ AD SOC

▸ AD SOC61

▸ LDAP simple TEST

▸ LDAP write TEST

▸ MS test

▸ AD test

▸ New Entry

Delete users with server:

No ▾

Allow user repository to change:

Yes ▾

Server to use to attempt to authenticate non-users:

---ANY--- ▾

Apply

Reset

On this page you define the repositories that will be used from this Swivel server to import users. It is not necessary to define every repository on every Swivel server in a HA solution, if you do not intend to synchronize them from this server, or authenticate user passwords.

When you create a repository, you need just two things: a name and a type. The repository types were defined above. The name can be anything, but each repository name must be unique, and if the repository name exactly matches an Agent name, that Agent can be used to manipulate users within that repository (only on the Swivel server). Once the repository name and type are entered, a new entry appears in the left menu, for you to enter the repository details. The required details for each repository type are listed below.

In addition to the list of repository servers, there are the following settings:

- **Delete users with server:** If **Yes**, then when you delete a server from this menu, all the users belonging to the repository will also be deleted. If **No**, then the users will not be deleted. Strictly, the repository name in the database is not deleted either - only the definition. This means that you can create a new repository with the same name as the deleted one, and the existing users will automatically be associated with it.
- **Allow user repository to change:** If **No**, and during a User Sync, a username is found that already matches a user in the Swivel database, an error is reported. If **Yes**, the existing user's repository is changed to the repository that is being Synced.
- **Server to use to attempt to authenticate non-users:** this option works together with the features of AgentXML and RADIUS that allow unknown users (users not in the Swivel database) to be authenticated using just their password. This option defines which repository is used to authenticate unknown users.

Types

Status

Log Viewer

▸ Server

▸ Policy

▸ Logging

▸ Messaging

▸ Database

▸ Mode

▾ Repository

▸ Servers

▸ Types

▸ Groups

▸ Attributes

→ Local

→ AD SOC

→ AD SOC61

→ LDAP simple TEST

→ LDAP write TEST

→ MS test

→ AD test

▸ RADIUS

▸ Migration

▸ Appliance

▸ OATH

▸ Config Sync

▸ Reporting

User Administration

Save Configuration

Repository / Types ?

Please configure the user repository classes.

Types:

▸ XML

▸ Active Directory

▸ Database

▸ Simple LDAP

▸ ADAM

▸ LDAP Writeable

▸ New Entry

Apply

Reset

This page is provided largely for information. It lists the available repository types. In theory, it is possible to add new repository types to this menu, but you would need to create new Java classes that implement the Swivel Repository interfaces.

Groups

Repository / Groups ?

Please enter the repository group information to be used by the Sentry server.
This includes group privileges and Active Directory/LDAP definition. For XML repository, please copy the group name into the definition.

	Single	Dual	Push	Mobile App	Admin	Helpdesk	PINless
Name: <input type="text" value="SwivelSingle"/> Definitions: Local: <input type="text" value="SwivelSingle"/> AD SOC: <input type="text"/> AD SOC61: <input type="text"/> LDAP simple TEST: <input type="text"/> LDAP write TEST: <input type="text"/> MS test: <input type="text"/> AD test: <input type="text"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Name: <input type="text" value="SwivelAdmin"/> Definitions: Local: <input type="text" value="SwivelAdmin"/> AD SOC: <input type="text"/> AD SOC61: <input type="text"/> LDAP simple TEST: <input type="text"/> LDAP write TEST: <input type="text"/> MS test: <input type="text"/> AD test: <input type="text"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Name: <input type="text" value="SwivelHelpDesk"/> Definitions: Local: <input type="text" value="SwivelHelpDesk"/> AD SOC: <input type="text"/> AD SOC61: <input type="text"/> LDAP simple TEST: <input type="text"/> LDAP write TEST: <input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Groups are used within Swivel to manage user rights, messaging and other features. A Swivel group can include users from multiple repositories, and each user can be a member of multiple groups. However, if a user is a member of more than one group that is used for messaging, and the message transports conflict, results can be unpredictable, so this is not recommended.

This page maps repository groups to Swivel groups, and assigns rights to each group.

The rights that can be assigned to each group are:

- Single - users in these groups can use single channel authentication methods
- Dual - users in these groups can use dual channel authentication methods (email or SMS)
- Push - users in these groups can use Push Notification to authenticate
- Mobile App - users in these groups can use the Swivel Mobile App to authenticate (not including OATH)
- OATH - users in these groups can use OATH tokens or soft tokens (e.g. Mobile App) to authenticate
- Admin - users in these groups can act as administrators
- Helpdesk - users in these groups can act as helpdesk users
- PINless - users in these groups are PINless (though may have a PIN for single channel, depending on policy)

Below each group name is a list of definitions that map the repository groups to the Swivel groups. For LDAP repositories, you can click the **Browse** button to the right to browse the repository to locate the group, rather than entering the full group name manually. group names for XML and Database repositories must be entered manually.

Name:

Definitions:


Local XML:

ActiveDirectory:

Apply Reset Group Rights

As well as the usual **Apply** and **Reset** buttons, this page also has a **Group Rights** button. This provides access to the **Group Rights** page, which is described below.

Groups Rights


swivelsecure

- Status
- Log Viewer
- Server
- Policy
- Logging
- Messaging
- Database
- Mode
- Repository
 - Servers
 - Types
 - Groups**
 - Attributes
 - Local
 - AD SOC
 - AD SOC61
 - LDAP simple TEST
 - LDAP write TEST
 - MS test
 - AD test
- RADIUS
- Migration
- Appliance
- OATH

Type: Helpdesk users normal users only ▼

User Groups	Helpdesk Groups
	SwivelHelpDesk
SwivelSingle	<input checked="" type="checkbox"/>
SwivelHelpDesk	<input type="checkbox"/>
SwivelMobile	<input checked="" type="checkbox"/>
SwivelToken	<input checked="" type="checkbox"/>
SwivelSMS_Strings	<input checked="" type="checkbox"/>
SwivelSMTP_Alerts	<input checked="" type="checkbox"/>
PINless	<input checked="" type="checkbox"/>
AD Container	<input checked="" type="checkbox"/>
SwivelSMTP_Strings	<input checked="" type="checkbox"/>
SwivelSMS_Alerts	<input checked="" type="checkbox"/>
SwivelDual	<input checked="" type="checkbox"/>
SwivelSMSPacketMedia_Strings	<input checked="" type="checkbox"/>
SwivelSMSPacketMedia_Alerts	<input checked="" type="checkbox"/>

Apply Reset

This page gives you fine control over managing which helpdesk groups can manage which user groups. Note that the policies defined here do not override the policy that helpdesk users can only manage users in their own repository, if set. Administrators can always manage all users, including other administrators. Helpdesk users cannot be configured to manage administrators.

There are 4 options listed in the **Type** drop-down:

- **Admin users only** - if this option is selected, helpdesk users have no special rights to manage other users. Only administrators can manage users.
- **Helpdesk users all groups** - this is the default, and gives all helpdesk groups the right to manage all other users, with the exception of administrators.
- **Helpdesk users normal users only** - as above, except that helpdesk users cannot manage other helpdesk users.
- **Helpdesk groups as below** - if this option is enabled, then the helpdesk groups matrix shown below the drop-down comes into play. For the other 3 options, that table is irrelevant.

The screen above shows just one helpdesk group, but if there are multiple groups with helpdesk rights, all helpdesk groups will be listed along the top in the column headers. All groups, except for those with administrator rights, are listed in the row headers on the left. Put a check in each cell if you want the helpdesk group at the top of the column to be allowed to manage users in the group to the left of the row, and leave the cell blank if you don't.

You will notice, if you select a type other than **Helpdesk groups as below**, that the check marks are automatically altered to match the chosen option:

- **Admin users only** - all checkboxes are cleared.
- **Helpdesk users all groups** - all checkboxes are checked.
- **Helpdesk users normal users only** - all helpdesk group checkboxes are cleared; all other groups are checked.

Attributes

The screenshot shows the 'Repository / Attributes' configuration page in SwivelSecure. The left sidebar contains a navigation menu with options like Status, Log Viewer, Server, Policy, Logging, Messaging, Database, Mode, Repository, Servers, Types, Groups, Attributes, Local, AD SOC, AD SOC61, LDAP simple TEST, LDAP write TEST, MS test, AD test, RADIUS, Migration, Appliance, OATH, Config Sync, Reporting, User Administration, Save Configuration, Upload Email Images, Administration Guide, and Logout. The main content area is titled 'Repository / Attributes' and includes a sub-header 'Please enter the repository attributes for the transport types (e.g. Email, Mobile Phone)'. There are two attribute entries displayed. The first entry is for 'email' and the second is for 'phone'. Each entry has a 'Name' field, a 'Phone Number?' dropdown, a 'Sync Rule' dropdown, an 'Add repository qualifier?' dropdown, and a list of attributes (Local, AD SOC, AD SOC61, LDAP simple TEST, LDAP write TEST, MS test, AD test) with input fields. A 'Delete' button is located next to each attribute list.

Swivel has the ability to store additional attributes against each user. These attributes can be used for the following purposes:

- As alternative usernames for authentication
- As message destinations - typically email addresses and phone numbers
- To search the user list in User Administration


This page allows you to define the attributes that are used, and how they are read from the repositories. A standard list of common attributes is provided initially, which can be modified as needed.

Each attribute has the following settings:

- **Name:** this is the name by which it is known within the Swivel application
- **Phone Number?:** if **Yes**, then the attribute is subjected to certain modifications when imported from the repository, as described later
- **Sync Rule:** Can be
 - ♦ **Synchronised** (the default), in which case it is always read from the repository;
 - ♦ **Initialised**, in which case it is only read from the repository for new users - if an attribute is modified in the repository later, the modification is not imported to Swivel;
 - ♦ **Local**, in which case the attribute is never imported from the repository - it can only be set using API calls.
- **Add repository qualifier?:** Can be
 - ♦ **None** (the default), in which case no qualifier is added to the attribute
 - ♦ **As prefix**, in which case the repository qualifier is prefixed to the attribute on import. The most common use for this is to form the Windows Account Name from Active Directory, as *domain\username*.
 - ♦ **As suffix**, in which case the repository qualifier is appended to the attribute on import. This is provided for completeness - it is unlikely to be used.

There then follows a list of repositories. Next to each repository, you should enter the repository attribute name that should be imported to the corresponding Swivel attribute. If the field is blank, nothing is imported to the attribute from that repository.

XML

 swivelsecure

Status

Log Viewer

Server

Policy

Logging

Messaging

Database

Mode

Repository

Servers

Types

Groups

Attributes

Local

AD SOC

AD SOC61

LDAP simple TEST

LDAP write TEST

MS test

AD test

RADIUS

Migration

Appliance

OATH

Config Sync

Repository / Local ?

Please enter the details for the XML repository.

Filename:

repository

Synchronization schedule:

NEVER ▼

Mark missing users as deleted:

Yes ▼

Initial PIN attribute:

pin

Initial Password attribute:

password

Import disabled users:

No ▼

Import disabled state:

No ▼

Reformat Phone Number:

No ▼

Prefix to remove:

Prefix to add:

Add domain qualifier:

None ▼

Repository Domain Qualifier:

Apply

Reset

An XML repository is maintained within the Swivel server itself, as an XML file. It is a writeable repository, meaning that users can be added and removed within the Swivel User Administration console.

There are two typical scenarios:

1. The XML repository contains a single user, admin, which is an administrator account. All other users are imported from Active Directory or some other external repository.
2. All users are managed within the XML repository. This scenario is typically for small installations that do not have an external directory, or do not wish to use it.

One thing to be aware of, for a HA environment, is that the XML file is not replicated over multiple servers. Therefore, if you want to maintain an XML repository over multiple servers, you must add users to every instance, or else manually copy the XML file to each server. Another solution that has been used is to have a different XML repository on each server, with different users. Be aware, however, that although XML repository users from one server cannot be managed on another server, they still exist on the other server, and have the same rights.

The following settings apply to XML repositories. All but the first are common to all repositories, and will not be described again in this document.

- **Filename:** The name of the XML file used to store users in this repository. This name will be generated automatically when the repository is created, so there should be no reason to change it. Be aware that changing the filename will result in existing users being lost, and that if two XML repositories have the same name, they will effectively be the same repository.
- **Synchronization schedule:** Synchronization schedule for the synchronization of this repository with Swivel. You configure here how often changes to the repository are updated to the Swivel database. Typically for the XML and other editable repositories this will be set to NEVER, as editable repositories are synchronized automatically when anything is changed from the Swivel admin console. However, for non-editable repositories, such as Active Directory, you should set this to an appropriate value depending on how often the repository changes. Typically, this will be once a day for infrequent changes, or once an hour for repositories that change frequently. Be aware, however, that for very large repositories, particularly if secure LDAP is used, the User Sync can take more than an hour. In the case that a user sync is still running when a new one is requested, whether manual or scheduled, the second request will be ignored.
- **Mark missing users as deleted:** If this is set to **Yes**, then when a sync job runs and an existing Swivel account is not found within the repository, rather than immediately deleting the account, the account is disabled and marked as deleted. The account can then be subsequently purged (deleted permanently) or undeleted. An account that is marked as deleted that is then detected during a subsequent sync job will be re-enabled. Note that accounts marked as deleted still count towards the total number of users for licensing purposes. The main advantage to using this option is that if users are deleted in error, their credentials are unchanged when they are restored. Generally, we always recommend that this is set to **Yes**.
- **Initial PIN attribute:** Name of the repository attribute that stores the initial PIN for the user. Can be used for manually setting specific initial PINs. Rather than setting the attribute name, it is also possible to set a fixed initial PIN for all new users by prefixing the value here with a # symbol, for example #1234 sets the initial PIN for all new users to 1234. This option is provided due to pressure from some customers. Swivel Secure does not recommend that you use it, as having the initial PIN for all users set to the same value is a security risk.
- **Initial password attribute:** Repository attribute that stores the initial password for the user. Can be used for manually setting specific initial passwords. As with initial PIN, this can be set to a fixed value, rather than reading from the repository, and as before, this is not recommended for security reasons. Be aware that this is a Swivel password, not a repository password. Setting a Swivel password is an additional security feature, but not all of our integrations support it: many assume the Swivel password is empty, and rely on the target system having its own password. An alternative in some scenarios is that Swivel can be configured to check the repository password.
- **Import disabled users:** If this option is set to **No** then users marked as disabled in the repository will not be imported into Swivel at all.
- **Import Disabled State:** Enable/disable the importing of users' disabled state from the user repository. When enabled the user repository will be consulted as to whether or not an account is disabled. Currently, Active Directory supports this functionality. Simple LDAP will support it if the name of the disabled attribute is entered in the appropriate settings. When enabled, it will no longer be possible to manually set the disabled state of the user within the Swivel administration interface. If **Import disabled users** is set to **No**, then this option has no effect, as disabled users will not be imported at all.
- **Reformat Phone Number:** If this option is set to Yes, then any phone number imported from the repository is reformatted by removing all non-digits (including spaces), and removing or adding a prefix, according to the following two options.
- **Prefix to remove:** If Reformat Phone Number is enabled and this option is not empty, then the first occurrence of the specified prefix is removed.
- **Prefix to add:** If Reformat Phone Number is enabled and this option is not empty, then the value of this option is added to the beginning of the number. A typical example of usage for phone number reformatting, in the UK, would be to set Prefix to remove to "0" and Prefix to add to "+44". This will ensure that phone numbers imported as, for example, "01937 582 020" will be stored in Swivel as "+441937582020". To avoid problems with international numbers, this prefix will be only be added if the prefix to remove was matched.
- **Add domain qualifier:** This option and the next allow you to add a fixed prefix or suffix to all usernames in this repository. This option specifies whether it should be a prefix, a suffix or neither. The prefix or suffix can be used to ensure uniqueness where there is a danger of having the same username in multiple repositories. It can also be used to ensure (for example in Active Directory) that the format of the username is correct for the target authentication platform. Be aware that if a prefix or suffix is used, users must always use them when authenticating to Swivel.
- **Repository Domain Qualifier:** This option allows you to specify what prefix or suffix should be added to users in this repository, as described in the previous option. If you use this option, make sure that any separator characters are included. For example, if usernames should be in the form *domainusername*, the prefix should be *domain*. Note that, as described in the [Attributes](#) section, rather than applying the prefix to the main username, it can also be applied to another attribute.

A Note on Attributes in XML Repositories

The XML repository uses a fixed schema for the XML file. This means that only certain values can be used as attribute names. These are

- username
- first-name
- last-name
- email
- phone
- password
- pin
- custom
- custom2
- custom3
- custom4
- custom5

Active Directory

Repository / AD SOC ?

Please enter the details for accessing Active Directory.

Hostname/IP:	<input type="text"/>
Username:	<input type="text"/>
Password:	<input type="password"/>
Port:	389 (Domain LDAP) ▼
Allow self-signed certificates:	Yes ▼
Synchronization schedule:	NEVER ▼
Username attribute:	sAMAccountName
Mark missing users as deleted:	Yes ▼
Initial PIN attribute:	<input type="text"/>
Initial password attribute:	<input type="text"/>
Import disabled users:	No ▼
Import disabled state:	No ▼
Ignore FQ name changes:	Yes ▼
Reformat Phone Number:	No ▼
Prefix to remove:	<input type="text"/>
Prefix to add:	<input type="text"/>
Add domain qualifier:	None ▼
Repository Domain Qualifier:	<input type="text"/>
Allow expired passwords:	No ▼
Attributes per batch:	1000
User container FQDN:	<input type="text"/>
Group container FQDN:	<input type="text"/>

This is the most commonly-used repository type. It allows the importing of user details from a Microsoft Windows Active Directory instance.


Active Directory has many of the same configurable options as an XML repository. It has the following additional options.

- **Hostname/IP:** Hostname or IP address of Active Directory server. For secure LDAP (port 636 or 3269), this MUST be the hostname matching the certificate, unless the certificate also has a Subject Alternate Name (SAN) for the IP address.
- **Username:** Username with appropriate permissions to connect to the repository. This should be a domain qualified username such as *user@swivelsecure.com* or *SWIVELSECURE\user*. For other LDAP-based repositories, this will typically be a fully-qualified distinguished name, for example *cn=user,dc=swivel,dc=local*.
- **Password:** Password for the user given above.
- **Port:** Port number used to connect to Active Directory:
 - ♦ **389 (Domain LDAP):** Connect to Active Directory's standard domain level LDAP interface.
 - ♦ **636 (Domain LDAP SSL):** Connect to Active Directory's standard domain level LDAP interface via an SSL-secured connection. This option should be used if the network connection between Swivel and the Active Directory server is untrusted. Be aware, however, that it is significantly slower than non-SSL connections.
 - ♦ **3268 (Global Catalog LDAP):** Connect to Active Directory's Global Catalog, allowing user searches to take place across domains in the forest.
 - ♦ **3269 (Global Catalog LDAP SSL):** Connect to Active Directory's Global Catalog via an SSL-secured connection, allowing user searches to take place across domains in the forest. This option should be used if the network connection between Swivel and the Active Directory server is untrusted. Be aware, however, that it is significantly slower than non-SSL connections.
- **Allow self-signed certificates:** Enable/disable the ability to connect via SSL to an Active Directory server whose certificate is not signed by a recognized certificate authority (e.g. if it is self-signed). This option also ignores the validity date of the certificate, but NOT the subject / hostname.
- **Username Attribute:** Repository attribute that stores the username a user will use to login. For Active Directory this could be **userPrincipalName** for a qualified username such as *user@swivelsecure.com*, or **sAMAccountName** for the pre-Windows 2000 name such as *user*. For other LDAP repositories, the attribute is typically **uid**.
- **Ignore FQ Name changes:** LDAP repositories such as Active Directory have a concept of fully-qualified names and account names. The fully-qualified name uniquely identifies the object within the repository and the account name is an attribute, such as **sAMAccountName**, that

the user then uses as a username. For example the fully qualified name may be *CN=test, CN=User, OU=IT, DC=swivel, DC=com*, but the Swivel account name will be created using the *sAMAccountName*, *test*. If the account is moved within the repository, the fully qualified name changes, e.g. *CN=test, CN=User, OU=Admin, DC=swivel, DC=com*. If you set **Ignore FQ name changes** to **Yes**, Swivel will ignore this change and the associated Swivel account will not be modified. If **Ignore FQ name change** is set to **No**, then the existing Swivel account will be deleted (or marked as deleted) and a new Swivel account will be created for that user, or else a duplicate user error will be reported.

- **Allow expired password**: This option allows you to specify if expired passwords or passwords that must be reset are allowed on the authentication.
- **Attributes per batch**: This option is currently not used.

ADAM

 swivelsecure

Swivel v4.0.3

Swivel A

- [Status](#)
- [Log Viewer](#)
- ⊕ Server
- ⊕ Policy
- ⊕ Logging
- ⊕ Messaging
- ⊕ Database
- ⊕ Mode
- ⊖ Repository
 - [Servers](#)
 - [Types](#)
 - [Groups](#)
 - [Attributes](#)
 - [Local XML](#)
 - [ActiveDirectory](#)
 - [ADAM](#)
- ⊕ RADIUS
- ⊕ Migration
- ⊕ Appliance
- ⊕ OATH
- ⊕ Config Sync
- ⊕ Reporting
- [User Administration](#)
- [Save Configuration](#)
- [Upload Email Images](#)
- [Administration Guide](#)
- [Logout](#)

Repository>ADAM

Please enter the ADAM configuration details.

Hostname/IP:	<input type="text"/>
Username:	<input type="text"/>
Password:	<input type="password"/>
Port:	<input type="text" value="389"/>
Use SSL:	<input type="text" value="SSL Off"/>
Synchronization schedule:	<input type="text" value="NEVER"/>
Username attribute:	<input type="text" value="uid"/>
Mark missing users as deleted:	<input type="text" value="Yes"/>
Initial PIN attribute:	<input type="text"/>
Initial password attribute:	<input type="text"/>
Import disabled users:	<input type="text" value="No"/>
Import disabled state:	<input type="text" value="No"/>
Ignore FQ name changes:	<input type="text" value="Yes"/>

- **Port**: For LDAP repositories other than Active Directory, the port is entered as a number.
- **Use SSL**: For Active Directory, SSL is automatically enabled for ports 636 and 3269, and disabled for 389 and 3268. For other LDAP repositories, you can manually specify whether or not to use SSL, and if enabled, whether to allow self-signed certificates (more generally, to ignore certificate validation) with this option.

An ADAM repository is very similar to Active Directory, with the additional ability of being able to create new users and modify existing ones directly in the Swivel admin console. Additional options here support that ability.

Reformat Phone Number:	<input type="text" value="No"/>
Prefix to remove:	<input type="text"/>
Prefix to add:	<input type="text"/>
Add domain qualifier:	<input type="text" value="None"/>
Repository Domain Qualifier:	<input type="text"/>
Allow expired passwords:	<input type="text" value="No"/>
Attributes per batch:	<input type="text" value="1000"/>
First name attribute:	<input type="text" value="givenName"/>
Surname attribute:	<input type="text" value="sn"/>
Group name attribute:	<input type="text" value="name"/>
User container FQDN:	<input type="text"/> <input type="button" value="Browse..."/>
Group container FQDN:	<input type="text"/> <input type="button" value="Browse..."/>
Expiry Date Attribute:	<input type="text" value="accountExpires"/>
New user password:	<input type="text" value="No password"/>
<input type="button" value="Apply"/> <input type="button" value="Reset"/> <input type="button" value="Browse in window"/>	

- **First name attribute** and **Surname attribute** are no longer relevant, as these can be specified under [Attributes](#).
- **Group name attribute**: the attribute used to identify a group within ADAM.
- **User Container FQDN**: the fully-qualified name of the container in which users will be stored within the repository. Users may be stored directly within here, or in sub-containers of this container. It is possible to enter this directly, but it is easier to browse for it, as long as you have already entered the server details. You can create a new container while browsing.
- **Group Container FQDN**: the fully-qualified name of the container in which groups will be stored within the repository. The same comments apply as for user container.
- **Expiry Date Attribute**: the attribute used to store the account expiry date. This is an optional feature. You can specify an expiry date for any writeable repository and the account will automatically be deleted (or marked as deleted) when that date occurs.
- **New user password**: You can specify whether or not to set a password for new users, and if a password is set, whether it is set manually or randomly generated. The repository password is used to authenticate to LDAP, and is distinct from the Swivel password. If you are only using the LDAP repository for Swivel, you do not need to know what the password is. However, be aware that in ADAM, the default policy requires that user accounts cannot be active unless they have a password. This means that users with no password will always be disabled in ADAM, so if you are importing the disabled attribute, they will be disabled in Swivel. The randomly generated password option is useful in this case: you do not need to know what the password is, but the user must have one.

Simple LDAP

Repository / LDAP simple TEST ?

Please enter the Simple LDAP configuration details.

User Container:	<input type="text"/>
Group Container:	<input type="text"/>
Administrator:	<input type="text"/>
Password:	<input type="password"/>
Server:	<input type="text"/>
Port:	<input type="text" value="389"/>
Base DN:	<input type="text"/>
Use SSL:	SSL Off ▼
Synchronization schedule:	NEVER ▼
Username attribute:	uid
Mark missing users as deleted:	Yes ▼
Initial PIN attribute:	<input type="text"/>
Initial password attribute:	<input type="text"/>
Import disabled users:	No ▼
Import disabled state:	No ▼
Base Search Context:	<input type="text"/>
Group ObjectClass Name:	groupOfNames
User ObjectClass Name:	inetOrgPerson
Member attribute name:	member
Member group attribute name:	<input type="text"/>
Ignore FQ name changes:	Yes ▼
User disabled flag name:	<input type="text"/>
User enabled flag name:	<input type="text"/>
Reformat Phone Number:	No ▼
Prefix to remove:	<input type="text"/>
Prefix to add:	<input type="text"/>
Add domain qualifier:	None ▼
Repository Domain Qualifier:	<input type="text"/>

A **Simple LDAP** repository has the same configurable items as an Active Directory repository but with the following additions:

- **Base DN:** the base distinguished LDAP name for the repository. Typically, this can be left blank unless doing so causes problems accessing the repository.
- **Base Search Context:** the base DN used when searching the repository. Again, this can normally be left blank unless you have difficulty synchronizing the repository.
- **Group ObjectClass Name:** the **objectClass** attribute value to be used for groups. Only groups with this object class will be searched when synchronizing. For writeable LDAP repository, this is the **objectClass** that will be used when creating new groups. It must therefore be a valid LDAP **objectClass**. Required parent classes will automatically be added.
- **User ObjectClass Name:** the **objectClass** attribute value to be used for users. The same comments apply as for **Group ObjectClass**.
- **Member attribute name:** the attribute used when locating or setting group membership for a user.
- **Member group attribute name:** the attribute used when locating group membership for a sub-group. Typically, this need not be set, as it is the same as for users, but some LDAP implementations use a different attribute.

- **User disabled flag name:** the attribute used to indicate that a user account is disabled. This is an optional attribute: if empty, all users are treated as enabled. Values must be of boolean type.
- **User enabled flag name:** the attribute used to indicate that a user account is enabled. This has the same use as the User disabled flag name, but with opposite logic: true indicates that the account is enabled, rather than disabled. No repository has been encountered so far that requires this attribute, but it is provided in the event that such a repository occurs.

LDAP Writeable

Repository / LDAP write TEST ?

Please enter the LDAP Writeable configuration details.

User container FQDN:	<input type="text"/>
Group container FQDN:	<input type="text"/>
Username:	<input type="text"/>
Password:	<input type="password"/>
Hostname/IP:	<input type="text"/>
Port:	<input type="text" value="389"/>
Base DN:	<input type="text"/>
Use SSL:	<input type="text" value="SSL Off"/>
Synchronization schedule:	<input type="text" value="NEVER"/>
Username attribute:	<input type="text" value="uid"/>
Mark missing users as deleted:	<input type="text" value="Yes"/>
Initial PIN attribute:	<input type="text"/>
Initial password attribute:	<input type="text"/>
Import disabled users:	<input type="text" value="No"/>
Import disabled state:	<input type="text" value="No"/>
Base Search Context:	<input type="text"/>
Group ObjectClass Name:	<input type="text" value="groupOfNames"/>
User ObjectClass Name:	<input type="text" value="inetOrgPerson"/>
Member attribute name:	<input type="text" value="member"/>
Member group attribute name:	<input type="text"/>
Ignore FQ name changes:	<input type="text" value="Yes"/>
User disabled flag name:	<input type="text"/>
User enabled flag name:	<input type="text"/>
Reformat Phone Number:	<input type="text" value="No"/>
Prefix to remove:	<input type="text"/>
Prefix to add:	<input type="text"/>
Add domain qualifier:	<input type="text" value="None"/>
Repository Domain Qualifier:	<input type="text"/>
First name attribute:	<input type="text" value="givenName"/>
Surname attribute:	<input type="text" value="sn"/>
Group name attribute:	<input type="text" value="name"/>
Expiry Date Attribute:	<input type="text" value="accountExpires"/>
New user password:	<input type="text" value="No password"/>
Membership attribute name:	<input type="text"/>
Password attribute name:	<input type="text" value="userPassword"/>
Container objectClass:	<input type="text" value="organizationalUnit"/>
Container attribute name:	<input type="text" value="ou"/>

This type of repository is a writeable version of Simple LDAP. As such, it has the configuration items of Simple LDAP, plus some others in common with ADAM, and the following additions:

- **Membership attribute name:** the attribute on a user account indicating membership of specific groups.
- **Password attribute name:** the name of the attribute in which the account password is stored. Note that for LDAP writeable, passwords are stored in plain text.
- **Container objectClass:** the **objectClass** attribute value used when creating new container objects.
- **Container attribute name:** the name of the attribute in which the name of a new container object is stored.

Database

- [Status](#)
- [Log Viewer](#)
- ⊞ Server
- ⊞ Policy
- ⊞ Logging
- ⊞ Messaging
- ⊞ Database
- ⊞ Mode
- ⊞ Repository
 - [Servers](#)
 - [Types](#)
 - [Groups](#)
 - [Attributes](#)
 - [Local XML](#)
 - [ActiveDirectory](#)
 - [Database](#)
- ⊞ RADIUS
- ⊞ Migration
- ⊞ Appliance
- ⊞ OATH
- ⊞ Config Sync
- ⊞ Reporting
- [User Administration](#)
- [Save Configuration](#)
- [Upload Email Images](#)
- [Administration Guide](#)
- [Logout](#)

Repository>Database

Please enter the details for the database repository

JDBC Driver:

Database URL:

Database login
user:

Database login
password:

Synchronisation
schedule:

User details table:

User ID field:

Username field:

Mark users as
deleted:

Initial PIN field:

Initial password
field:

Import disabled
users:

Import disabled
state:

Disabled flag field:	<input type="text"/>
Group membership table:	<input type="text"/>
Membership userid field:	<input type="text"/>
Membership group field:	<input type="text"/>
Add domain qualifier:	<input type="text" value="None"/>
Repository Domain Qualifier:	<input type="text"/>
Reformat Phone Number:	<input type="text" value="No"/>
Prefix to remove:	<input type="text"/>
Prefix to add:	<input type="text"/>

Database repositories are used where Swivel needs to read user information from an existing database. At present, there is no writeable version of this repository. Some configurable items for this repository are common to XML repositories. The following are the additional configurable items:

- **JDBC Driver:** the full class name for the JDBC driver class used to access this database. Swivel can only access databases through JDBC drivers (the same as for Swivel's own database), so the appropriate JDBC driver jar file must be copied to the sentry/WEB-INF/lib or <tomcat_home>/lib folder.
- **Database URL:** the JDBC URL for the target database. This must include the database / namespace.
- **Database login user:** the username for the database account. This must be an account on the database server that has read rights to the appropriate database tables. Note that JDBC does not support Windows Integrated Authentication for MS SQL databases, only built-in database accounts.
- **Database login password:** the password for the above account.
- **User details table:** the name of the table containing usernames that are to be imported into Swivel.
- **User ID field:** the name of the field in the users table containing user IDs. See the next value for details.
- **Username field:** the name of the field in the users table containing usernames as Swivel will use them. The user ID and username field can be the same field, and typically they will be, but if the database contains distinct unique fields, you can specify different fields.
- **Group membership table:** the name of the table containing group membership information. This table must contain at least the user ID and a group name column. If you already have such a table, then fine. Alternatively, you may find that the users table contains a category column which matches the required group membership information, in which case the group membership table can be the same as the users table. Failing that, if you will only be using one group in Swivel, and all users will be in that group, or you can write a SQL query to select the members of that group, then you can create a View in the database to represent the list of User IDs required, with a constant value for the group name.
- **Membership userid field:** the field in the group membership table containing the user ID. Note that this must correspond to the user ID value in the users table, not the username value, if they are different.
- **Membership group field:** the field in the group membership table containing the group name. The values in this table must correspond to values in the Repository Groups definitions for this repository.

A Note on Attributes

Attributes as defined in the [Attributes](#) section for the database repository must be column names from the users table. If you do not have a single table that contains all the required information, you will need to define the users table using a View.