



# Swivel Reference Manual Version 3.9

# Contents

Introduction.....	5
Getting Started with Swivel 3.9.....	6
Setting the Swivel Database.....	6
Setting the Internal Database .....	7
Setting the Database Mode.....	7
Configuring Slave Mode .....	7
Configuring Synchronized Mode .....	7
Setting up user repositories.....	7
Using the XML Repository .....	8
License Key .....	9
Integration .....	10
Integrating with an external database .....	10
Integrating with user repositories .....	11
Using Groups with User Repositories.....	11
Importing Disabled State .....	13
Ignoring Fully Qualified (FQ) name changes .....	13
Marking missing users as deleted.....	13
Considerations for multiple repositories.....	14
Configuring the XML Repository .....	14
Configuring an Active Directory Repository .....	14
Configuring other LDAP Directory Servers as Repositories.....	20
Other repository types.....	23
Deleting a Repository .....	23
Integrating with Transport Classes.....	24
Configuring transports in shared database Swivel installations .....	25
How to integrate Swivel with a GSM Modem.....	25
How to integrate Swivel with an SMS service Provider.....	26
Integrating via SMTP .....	27
Integrating with Agents using Agent-XML.....	27
How to integrate Swivel with an external agent .....	27
Using the User ChangePIN application .....	28
How to integrate Swivel with an IIS Website.....	30
How to write an Agent.....	32

How to use the custom attribute .....	32
Integrating using RADIUS.....	32
RADIUS Groups .....	34
How to integrate a VPN with Swivel using RADIUS.....	35
Two Stage RADIUS authentication.....	35
How to use single channel with a VPN .....	35
SysTray Utility.....	36
How to incorporate a TURING image into a VPN log on page.....	38
Integrating with 3 <sup>rd</sup> Party Authentication systems.....	38
Integrating with other Authentication Servers.....	39
Peering Swivel servers .....	39
Swivel as a RADIUS proxy.....	41
Multiple Swivel servers using a single database .....	42
Operation & Maintenance.....	43
Tasks .....	43
Data Migration .....	45
Jobs .....	45
Session Clean Up.....	45
Peer Synchronization.....	46
Inactive User Check.....	46
PIN expiry check.....	46
Logs and Alarms .....	46
XML Logging .....	46
Syslog .....	48
SMTP (email) Logging.....	49
Administrator's Guide .....	50
Setting Policies .....	50
Standard.....	50
PINless.....	50
PIN and OTC Policies .....	52
Password Policies.....	54
User self-reset.....	54
User Policies .....	55
Single Channel.....	55
Dual Channel .....	56
Alerting Users .....	57
Managing Users .....	57
Adding User Groups.....	57

Adding Users.....	59
Appendix A: Building and Deploying Swivel server .....	63
Upgrading from Swivel version 3.2 onwards.....	63
Upgrading from Swivel version 3.1.....	64
Architecture .....	64
Swivel Deployment.....	65
Protecting the Swivel Admin Console.....	65
Configuring the filter .....	65
Editing filter.properties.....	65
Editing ranges.xml .....	66
Activating the Filter .....	67
Filter in operation.....	67
Session Sharing .....	67
Appendix B: Windows Installation of the Java Comm API .....	69
Testing.....	69
Appendix C: Active Directory/LDAP Groups and Attributes.....	70
User rights .....	70
Simple AD Group and Attribute example .....	70
More Complex AD Group and Attribute example.....	72
Appendix D: Swivel Installation details. ....	75
Appendix E: Setting Schedules and CRON Strings .....	76
Document Versioning .....	77

# Introduction

This document provides a general overview of Swivel version 3.9, its key features and a quick start guide to the Administration console. It also covers what you should know about your installation of Swivel and how to support and maintain your installation of Swivel.

The key new features are

1. Report Scheduling
2. Multiple writable repositories for OpenLDAP, ADAM and XML.
3. Improved LEAP support
4. SSL support for SMTP
5. Telephony capability using Asterisk
6. Improved transports model
7. Improved Federation (SAML 2.0) support

More details can be found within the associated Release Bulletin, available from [www.swivelsecure.com](http://www.swivelsecure.com)

Extra documentation can be found at the Swivel knowledgebase available at [kb.swivelsecure.com](http://kb.swivelsecure.com)

Note: An important part of this document is the record of the installation, covered in Appendix D: Swivel Installation details. It is recommended that this section is completed at the time of installation and kept with the server or elsewhere where it can be found easily if required. A copy of the installation record needs to be sent to Swivel Secure for their records; this can be sent to:

e-mail            [support@swivelsecure.com](mailto:support@swivelsecure.com)  
fax                +44 1937 547 590

## Getting Started with Swivel 3.9

This guide assumes that there is a Swivel Version 3.9 server installed and running. Information about building and deploying a Swivel server is available in .

This section follows the steps required to get a basic Swivel installation up and running.

There are three fundamental settings that need to be completed to get a Swivel server up and running.

1. The Swivel Database
2. The Database mode of operation
3. Any associated user repositories

### Setting the Swivel Database

Swivel needs a database to store Swivel account information. This can be an internal or an external database.

On install the Swivel server comes with a shipping database; this is a single user read-only database that has the user account:

Username: admin

PIN: 1234

Whilst this option is selected as the database it will only be possible to login to the admin console using these details.

Therefore to log in for the first time, enter admin in the username field, click on start session, then enter the first four digits of the TURING image into the OTC field, then click log-in. (The password field should be blank)

The first stage in getting started with Swivel is to configure the database that you wish to use to store Swivel account details. This can either be the internal database that comes with Swivel or it can be a separate external database. Swivel supports a range of SQL/JDBC databases: consult with your reseller or with Swivel for more details.

In deciding what database to use you need to evaluate the relative merits of the two approaches.

Factor	Internal Database	External Database
Simplicity	Very simple, single-click deployment.	Requires some database set-up and configuration
Flexibility	No Flexibility	Flexible solution allowing multiple Swivel and Multiple Database servers as required
Performance	Local database means good performance.	Performance needs to be considered but Swivel is not a particularly database intensive application
Availability	Availability determined by the server it is deployed on. Difficult to back-up database, therefore not easy to support site resilience	Available can be improved by using database clustering technologies. External database easier to back-up and replicate

Security	Internal and encrypted	All sensitive data is encrypted. Encrypted database drivers can also be used.
----------	------------------------	--

It is assumed that an external database will be more appropriate to larger, multi-site installations and the internal database for use for single site installations. For details integrating with an external database refer to .

### ***Username Case Sensitivity***

You can select whether usernames are case sensitive or not. You may need to set usernames to be case insensitive if that is what the users are used to (e.g. Windows usernames are not case sensitive). When user names are not case sensitive a user with a username of Chris can authenticate a Chris, chris,chrIS etc.

NOTE: Case sensitivity is also affected by settings in your database, if you use an external database. If you require case-sensitive passwords, you also need to ensure that your database is set to be case sensitive.

## **Setting the Internal Database**

To use Swivel's internal database go to the Database menu, select Internal from the drop down menu, then click apply. There may be a slight delay as Swivel creates the tables in the database. You then need to create an admin account, see Setting up user repository. If another database type is required refer to the section on Database Integration

## **Setting the Database Mode**

Swivel supports two different database modes; synchronized and slave. Synchronized will mean that the Swivel server will synchronize with a user repository (e.g. Active Directory) in order to create (or delete) accounts from the Swivel server. A Swivel server running in slave mode will not create or remove accounts from the database but will act as an authentication server for all the accounts that exist in the database. A slave Swivel server relies on another Swivel server to add and remove accounts.

### **Configuring Slave Mode**

In slave mode there is no user repository. When configuring Swivel to use slave mode it is therefore important that the database that you have defined has users (and admin users) already in it. Once you select the slave mode you should go to the user admin screen to ensure that the users exist on the server.

### **Configuring Synchronized Mode**

To use this mode go to the Mode->General screen and select Synchronized and then click apply.

The next stage is to configure the repository.

## **Setting up user repositories**

Swivel supports multiple user repositories. The initial repository is the internal XML repository, which can be edited within the Swivel Administration Console. Additional repositories can be defined, repositories can be LDAP based (eg Active Directory) or SQL based (eg MySql Database).

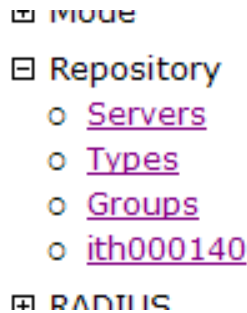
If you are using a user repository it is important that you create an Administrator user to prevent you being locked out of the admin console. If you are unfamiliar with Swivel, it is recommended that you use the internal XML repository initially.

## Using the XML Repository

To use the XML repository, you need to go to the Repository->Servers screen and add an XML repository. You need to give the XML repository a name. Repository names need to be unique within a Swivel installation; therefore if you have two Swivel servers connected to the same database, they cannot both have an XML repository called LOCAL.

The XML repository is also a special case because you can add/edit users for this repository from the Swivel user administration screen.

Go to Repository->Servers to confirm that this repository has been configured. It should be the first (initially, the only) repository in the list, and should appear in the menu below the Groups heading:



Go to Repository->Groups to configure the repository groups. Initially, two groups are defined – SwivelUsers and SwivelAdministrators. Make sure that each group definition for this repository is the same as the group name, as shown below:

### Repository>Groups

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

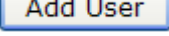
	Single	Dual	Swivlet	Admin	Helpdesk	PINless
Name: <input type="text" value="PINsafeUsers"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>Definitions:</b>						<input type="button" value="Delete"/>
PINsafe1: <input type="text" value="PINsafeUsers"/>						
Name: <input type="text" value="PINsafeAdministrators"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<b>Definitions:</b>						<input type="button" value="Delete"/>
PINsafe1: <input type="text" value="PINsafeAdministrators"/>						
Name: <input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>Definitions:</b>						
PINsafe1: <input type="text"/>						

Go to user administration screen and select

The XML repository is shipped with an existing user of admin, with a PIN of 1234, so you will see this account appear on the list of users.



It is recommended that you open a new browser window, navigate to the admin console and log on using this new account before you exit the existing admin console session.

You can create new Admin level accounts by selecting  on the User Administration screen and creating a new user that is a member of the Swivel Administrators group. You can then synchronise Swivel with the repository to create the Swivel account. Remember to reset the PIN of the new Admin user that you have created.

NOTE: Unlike previous versions of Swivel, this repository is always available, even when you add another repository. Therefore, you need to ensure that you change the PIN for the admin user, or else delete or disable this user.

## License Key

Swivel comes with a 5 user evaluation license. To operate a live Swivel server you need a valid licence key obtained from your reseller or from Swivel Secure. Once you have this license key enter this key on the Server -> License screen. The licence will be for a fixed number of users; i.e. accounts on the Swivel server. If you need additional users and therefore additional licences, you can purchase a new licence key for the new total of licences required. The new license key is a replacement for the existing one and therefore you simply need to overwrite the license key.

With the repository and database configured and the license installed you are now have a working Swivel server that you can start to integrate with your IT infrastructure.

## Integration

The integration tasks for Swivel depend on the specifics on the installation and the technology with which Swivel is required to operate. There are a number of potential integration points including:-

### ***Databases:***

Swivel can be configured to use an external database.

### ***User Repository:***

Swivel comes with its own user repository but it can also be configured to work with external user repositories such as Active Directory

### ***Agents:***

Agents are the integration points between Swivel and what it is that Swivel is being used to protect. Agents use Swivel's Agent-XML API to integrate with Swivel.

### ***Radius:***

Swivel also can act as a RADIUS server, so an alternative method of integration is to configure Swivel to accept authentication requests from a RADIUS (NAS) client.

### ***Third Party Authentication:***

Swivel is an open authentication platform in that it can be used in conjunction with other authentication technologies. Swivel has a third party API to support this interaction.

### ***Transport:***

Swivel sends security strings to end users via a transport layer implemented by a transport class. Different classes allow for security strings to be sent via different methods, e.g. SMTP, SMS. Transport classes can also be configured to send alert information to users, e.g. to inform them that their Swivel account has been created.

### ***Logs:***

Swivel can be configured to send logs to external systems including syslog and SMTP. These integration points are described in more details below.

## Integrating with an external database

Note: Swivel is shipped with the Swivel software required to interface with many popular databases but does not include any licences/drivers for those databases. It is up to the user to obtain any required database licences. Swivel supports opensource databases such as MySQL and PostGres. If you purchase Swivel as a HA Active-Active Appliance, MySql is pre-installed and is appropriately licensed.

This section describes the stages required to configure at Swivel to work with an external database. Most databases will require similar steps; we use MySQL5 as an example.

The first stage is to create a database that Swivel can use to create the tables it needs and store the data. This database needs to support the UTF8 character set.

You then need to ensure that Swivel can connect to that database, e.g. that all firewalls have the required port open (3306 being the default).

Obtain the required database drivers for the database.

Copy these drivers onto the Swivel server under the webapps/Swivel/WEB-INF/lib directory.

Restart the Swivel server

Now go to the Database configuration screen and enter the details of the database and the drivers that you have configured. See the example below.

Identifier:	<input type="text" value="MySQL 5"/>	
Class:	<input type="text" value="com.swiveltechnologies.user.database.MySQL5Dat"/>	
Driver:	<input type="text" value="com.mysql.jdbc.Driver"/>	
URL:	<input type="text" value="jdbc:mysql://192.168.0.156:3306/pinsafe"/>	
Username:	<input type="text" value="fred"/>	
Password:	<input type="password" value="••••"/>	<input type="button" value="Delete"/>

Once you have entered this information select your chosen database from the drop-down list, then click Apply. Swivel will now try and create the tables in the database. Check the log files to see if there are any errors. If, for any reason, Swivel could not open the new database, the database setting reverts to the existing setting to prevent the user being locked out.

If the database tables were created successfully you will see a Database Opened message.

Once the new database has been created, it is important to synchronize with the user repository to ensure that an admin account is created so that the admin console can be accessed. Remember to set the PIN of this account to a known value.

## Integrating with user repositories

Swivel comes with an XML repository that can be used to store user accounts. However, enterprises may already have a user repository, e.g. Active Directory, for user accounts, and this can be integrated with Swivel.

The integration means that Swivel will synchronise with the external user repository to ensure that the user accounts on Swivel match those within the external user repository. To ensure that the two data stores remain synchronized, Swivel can be configured to synchronize with the external repository at regular intervals, e.g. once an hour, or synchronisations can be instigated manually.

## Using Groups with User Repositories

Swivel grants users rights dependent on the groups to which they belong within the repository.

To configure groups, go to the Repository > Groups screen, will look something like the following:

## Repository>Groups

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

	Single	Dual	Swivlet	Admin	Helpdesk	PINless	
Name: <input type="text" value="PINsafeUsers"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<b>Definitions:</b>							<input type="button" value="Delete"/>
PINsafe1: <input type="text" value="PINsafeUsers"/>							
AD: <input type="text" value="CN=PINsafe,OU=Groups,DC=test,DC=local"/>							
Name: <input type="text" value="PINsafeAdministrators"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
<b>Definitions:</b>							<input type="button" value="Delete"/>
PINsafe1: <input type="text" value="PINsafeAdministrators"/>							
AD: <input type="text" value="CN=PINsafeAdmins,OU=Groups,DC=test,DC=local"/>							
Name: <input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<b>Definitions:</b>							
PINsafe1: <input type="text"/>							
AD: <input type="text"/>							
							<input type="button" value="Apply"/> <input type="button" value="Reset"/>

With a group name and then a definition for that group for each repository defined on the Swivel server

There are two parts to defining a new group: defining what rights the group has, and mapping the group to user groups within the repositories.

This screen defines six basic rights: (TBD AD Add Telephony)

- Single – connect using single-channel authentication
- Dual – connect using dual-channel authentication
- Telephony – use PSTNsafe server
- Mobile Client – use a mobile client application
- Admin – administer the entire Swivel configuration
- Helpdesk – administer Swivel users
- PINless – user has no PIN, just a password

When planning what groups are required, in addition to the above functions, the following need to be taken into account:

- Server Agents
- Transport classes
- RADIUS NAS
- RADIUS Groups
- Third Party authentication

When planning the groups, there is no need to define a different group for each function. If, for example, all dual-channel users use the same transport class, then the same group can be used for both functions. Also, users can be members of multiple groups. The exception to this is that a user cannot have more than one transport class.

Each group must have a unique name. Group names are just labels within the configuration, and should be descriptive of the group's purpose.

Having determined what groups are required, the next step is to define the mapping to the repositories.

This is slightly different for the XML repository than for Active Directory or LDAP. For Active Directory and LDAP repositories, the definition is the full-qualified domain name (FQDN) representing the user group within the repository. This group must exist within the repository: defining groups and group membership of Active Directory and LDAP repositories depends on the repository you are using, and is outside the scope of this document - see the appropriate documentation for your directory server. For the XML repository, the group definition is just a label for the user edit screen (and within the XML file). You would normally just copy the group name as the XML repository group definition.

It is not necessary to give a definition for every group for every repository. If a definition is left blank, then no users from the respective repository will be members of the respective group. This goes for the XML repository as well as other repositories.

Refer to Appendix B to see example Active Directory, LDAP and XML repository groups and attributes.

## Importing Disabled State

If, as is the case with Active Directory, the repository supports the principle of disabled accounts, Swivel can take account of this when synchronising with the repository. If Import disabled state is set to Yes, then if the account is disabled in the repository it will be disabled within Swivel. If Import disabled state is set to No, then a user account can be manually disabled from the User Administration screen by clicking on the user account, selecting policy and then selecting Disabled.

If Import Disabled state is set to Yes, it is not possible to manually disable accounts.

## Ignoring Fully Qualified (FQ) name changes

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, if the fully qualified account name changes but the account name remains the same 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 and a new Swivel account will be created for that user.

## Marking missing users as deleted

Swivel users group membership within the user-repository is used to determine the users rights within Swivel. If a user is removed from a group it is assumed that the user should be removed from Swivel. However it is recognized that the user (or even the whole group) may have been

removed in error. If this has happened the accounts will be lost and the only way to get them back would be to re-provision the accounts and allocate those users new PINs. Since version 3.5, it is now possible to not automatically delete users that are missing from the repository. If this option is chosen the missing account will not be deleted, but merely disabled. If, in subsequent jobs, the account re-appears, (the error being rectified), the account is re-enabled without the user needing to be assigned a new PIN. If there is a requirement to delete the account, it can be purged from the User List screen on the admin console.

## Considerations for multiple repositories

Swivel has the ability to support multiple repositories. If you are planning on using multiple repositories, you should be aware that usernames must be unique across all repositories. For example, it is not possible to have a user called admin in the XML repository and one also called admin in an Active Directory repository. If you are not going to be using the XML repository, it is advisable to remove all users from it, both from the point of view of security and of simplifying synchronization of the Active Directory. If it is going to be difficult to guarantee uniqueness of username across multiple repositories, it might be advisable to use email address, for example, as the login name instead.

## Configuring the XML Repository

A new installation of Swivel has a single repository, based on an XML file. It starts with a single user, “admin”, with a PIN of 1234. This user can be removed or disabled, but at the very least it is strongly recommended that the PIN is changed.

It is not normally necessary to modify the configuration of the XML repository, except possibly to change the synchronisation schedule, but the details for doing so are given here.

Select Repository, and then the name of the XML repository, which should be the name of the Swivel server.

The configuration screen should look like this:

**Repository>ptest** ⓘ

Please enter the details for the XML repository.

Username attribute:

PIN attribute:

Password attribute:

Synchronization schedule: Every  at  minutes past the hour

Username, initial PIN and Initial password are provided for compatibility with other repositories, but there is little point in changing these.

It may well be worthwhile, however, changing or removing the synchronization schedule. Since XML repository users can be added and edited within the administration console, it is easy to synchronize the repository manually after making changes. Having an automatic synchronization, therefore, serves little purpose. To remove it, set the schedule to never.

## Configuring an Active Directory Repository

Swivel provides a class for the integration of Swivel with an Active Directory. This class acts as an

interface between the Swivel and the external active directory user repository.

To set-up Swivel to integrate with Active Directory

1. Set up the required groups within Active Directory and add users to those groups. See Appendix C: Active Directory/LDAP Groups and **Attributes**.
2. Add the Active Directory server as a repository on Swivel
3. Configure the interface between Swivel and AD
4. Configure the group definitions for the repository
5. Synchronise the users into Swivel.

Note that the repository name and type cannot be changed once you have added the server - you must delete the server, and add a new one to rename it.

### **1 Set up Groups**

Groups need to be created in AD that represent different user roles within Swivel, e.g. users, helpdesk user, admin users etc. refer to Appendix C: Active Directory/LDAP Groups and **Attributes**

### **2 Add Repository**








Go the Repository > Servers screen. Add a new server by entering an appropriate name in the blank entry at the bottom of the list. From the drop down options select Active Directory, and then click Apply.

### **3 Configure Interface**

The next stage is to configure the connection to the AD server. In order to do this you must have an AD domain user account and password and the Swivel server must be able to access the server via the selected port.

## Repository>adtest

Please enter the details for accessing Active Directory.

Hostname/IP:	<input type="text" value="192.168.0.165"/>
Username:	<input type="text" value="Administrator@test.loc"/>
Password:	<input type="password" value="....."/>
Allow self-signed certificates:	<input type="button" value="No"/> 
Username attribute:	<input type="text" value="sAMAccountName"/>
PIN attribute:	<input type="text"/>
Password attribute:	<input type="text"/>
Import disabled state:	<input type="button" value="No"/> 
Ignore FQ name changes:	<input type="button" value="Yes"/> 
Mark missing users as deleted:	<input type="button" value="No"/> 
Port:	<input type="button" value="389 (Domain LDAP)"/> 
Synchronization schedule:	Every <input type="button" value="hour"/>  at <input type="button" value="30"/>  minutes past the hour
	<input type="button" value="Apply"/> <input type="button" value="Reset"/>



Select the repository name from the left hand menu and complete the form: The entries required are summarised in the table below:

Parameter	Meaning
Host/IP	The hostname or IP address of the Domain Controller that Swivel will connect to.
Username	This username needs to be a fully qualified username for a user within the Active Directory domain that has the required privileges.
Password	Input the password associated with the above account
Synchronization schedule	Set the synchronisation schedule. This determines how often Swivel will update its user list from Active Directory. See Appendix E: Setting Schedules and CRON Strings for details.
Allow self-signed certificates	See Port
Username attribute	Set the username attribute; this is the attribute within the Active Directory repository that will form the username within Swivel. The default for this is sAMAccountName. This is generally the most appropriate as it is the username that will be used for other, Windows based, authentication
Port	Select the required port number. If you are using SSL for the connection between Swivel and Active Directory you need to select Port 636 and decide whether to accept self-signed certificates. If you are connecting to a multi-domain AD installation you may need to use Global Catalogue LDAP (3268) or its SSL equivalent (3629).
Initial PIN/Password	Optionally set the attributes within active directory that will form in the initial values for password and PIN.
Import Disabled/Ignore FQDN	Set the flags to import disabled state and ignore FQ name changes as described earlier.
Mark missing users as deleted	If a user account no longer exists in the repository, do not delete the account but mark it as deleted. Accounts can be re-enabled or purged from the admin console.

Click  then go to the User Admin screen and click 

At this stage we have not defined what groups within active directory we are interested in so no users will be synched across. Go to the log viewer: if you have connected successfully you will see a message saying synch started and synch completed.

If this was not successful you will see an error in the log; for example

14:22:23 20 December r 2006	ERROR	192.168.0.13 admin: Exception occurred during repository attribute query, object: , attribute: defaultNamingContext, exception: javax.naming.AuthenticationException: [LDAP: error code 49 - 80090308: LdapErr: DSID-0C090334, comment: AcceptSecurityContext error, data 525, vece ]
--------------------------------------	-------	---

Indicates the username or password were not valid.

14:25:27 20 December r 2006	ERRO R	192.168.0.13 admin: Exception occured during repository attribute query, object: , attribute: defaultNamingContext, exception: javax.naming.CommunicationException: 192.168.0.199:389 [Root exception is java.net.NoRouteToHostException: No route to host]
--------------------------------------	-----------	--

Indicates the server could not be reached.

Once you have confirmed that you can connect to the AD server; you need to set up the repository groups.

## 4 Configure Groups

Go to Repository > Groups

Assuming you have not added new groups, there will be just the two groups defined: SwivelUsers and SwivelAdministrators. The definitions for the AD server will be blank. You can enter fully-qualified domain names for the appropriate groups in the Active Directory which represent normal users and administrators but it is recommended that the Browse feature is used.

### Repository>Groups

Please enter the repository group information to be used by the PINsafe server.

This includes group privileges and Active Directory/LDAP definition. For XML repository, please copy the group name into the definition.

	Single	Dual	Swivlet Admin	Helpdesk	PINless
Name: <input type="text" value="PINsafeUsers"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>Definitions:</b>					
xxxx: <input type="text" value="PINsafeUsers"/>	<input type="button" value="Delete"/>				
adtest: <input type="text" value="CN=pinsafeusers,OU=pinsafe,DC=test,DC=local"/>	<input type="button" value="Browse"/>				
	<input type="button" value="Browse"/>				

This feature allows the repository to be browsed and the required group to be selected.

## Repository name: adtest

**Path: OU=pinsafe,DC=test,DC=local**

### [Up a level](#)

#### Groups:

<a href="#">CN=dual</a>	Select
<a href="#">CN=helpdesk</a>	Select
<a href="#">CN=iTAGG</a>	Select
<a href="#">CN=pinless</a>	Select
<a href="#">CN=pinsafeadmin</a>	Select
<a href="#">CN=pinsafegroup1</a>	Select
<a href="#">CN=pinsafegroup2</a>	Select
<a href="#">CN=pinsafeusers</a>	Select
<a href="#">CN=single</a>	Select

Once the group has been selected go to User Administration and select

User Sync

This will then create Swivel accounts associated with the active directory accounts, including Swivel accounts with admin rights for active directory accounts that belong to the relevant group, if any. These accounts will be listed on the User Admin screen.

Assuming you have created an administrator account in this repository, select it. Select RESET PIN for the account and enter a new PIN. Make a note of this PIN.

The server is now configured to pull in user information from the Active Directory. There are two ways of instigating the pull of this data:

Manually                      By going to the User Administration screen and selecting User Sync.

Automatically                The schedule for a given repository is set on the configuration screen for that repository, as described above.

After you have synchronised with the AD repository you should see the accounts that have been created in the User Administration screen. **Remember to reset the PIN on any new admin accounts that you have created, if they are not created automatically.**

## Configuring other LDAP Directory Servers as Repositories

Swivel provides a class for the integration of Swivel with other LDAP directory servers. This has been tested for use with OpenLDAP, Sun Directory Server, Novell eDirectory and IBM Tivoli Directory Server, and will probably work with most other LDAP-compatible repositories that implements group membership according to the LDAP recommendations ([RFC 2256](#)).

The steps for other LDAP repositories are the same for Active Directory

To set-up Swivel to integrate with an LDAP repository

1. Set up the required groups within LDAP and add users to those groups. See Appendix C: Active Directory/LDAP Groups and **Attributes**.
2. Add the LDAP server as a repository on Swivel
3. Configure the interface between Swivel and LDAP
4. Configure the group definitions for the repository
5. Synchronise the users into Swivel.

Note that the repository name and type cannot be changed once you have added the server - you must delete the server, and add a new one to rename it.

### 1 Set up Groups

Groups need to be created in LDAP that represent different user roles within Swivel, e.g. users, helpdesk user, admin users etc. refer to Appendix C: Active Directory/LDAP Groups and **Attributes**


### 2 Add Repository

Go the Repository > Servers screen. Add a new server by entering an appropriate name in the blank entry at the bottom of the list. From the drop down options select Simple LDAP, and then click Apply.




### 3 Configure Interface

The next stage is to ensure that you can connect to the directory server. In order to do this you must have a valid user account and password for the directory server and the Swivel server must be able to access the server via LDAP (by default port 389, but most servers allow this to be changed on installation).

Enter the details of the directory server and the account you are using. Note that the username may need to be fully qualified.

**Repository>LDAP** 

Please enter the LDAP configuration details

Administrator:	<input type="text" value="cn=Admin,ou=test,o=lo"/>
Password:	<input type="password" value="•••••"/>
Server:	<input type="text" value="LDAPServer"/>
Port:	<input type="text" value="389"/>
Base DN:	<input type="text"/>
Synchronization schedule:	<input type="text" value="0 0 * * * ?"/>
Username attribute:	<input type="text" value="uid"/>
Initial PIN attribute:	<input type="text"/>
Initial password attribute:	<input type="text"/>
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"/>
Import disabled state:	<input type="text" value="No"/> 
Ignore FQ name changes:	<input type="text" value="No"/> 
User disabled flag name:	<input type="text" value="disabled"/>
User enabled flag name:	<input type="text"/>
Use SSL:	<input type="text" value="SSL Off"/> 
<input type="button" value="Apply"/> <input type="button" value="Reset"/>	

**Configuration for LDAP repository**

The values are explained in the following table:

Parameter	Meaning
Administrator	The distinguished name of the LDAP user credentials to access the server. This username needs to be a fully qualified username for a user within the LDAP server that has the required privileges
Password	The LDAP user's password
Server	The name or IP address of the server hosting the directory service
Port	The port on which LDAP is running
Base DN	The base distinguished name of the LDAP server
Synchronization Schedule	How often the Swivel repository is updated from the directory server (see Appendix E: Setting Schedules and CRON Strings for details on this)
Username attribute	The name of the attribute on a user object to use as the Swivel user name. The default for this is uid. This is the most appropriate attribute if users are implemented as inetOrgPerson objects, or custom extensions to that object class
Initial PIN attribute	The name of the attribute on a user object to use as the initial PIN for a user
Initial password attribute	The name of the attribute on a user object to use as the initial password for a user
Base Search Context	The sub-context on the LDAP server in which all users and groups exist
Group ObjectClass Name	The name of the LDAP object class representing a group. For most servers, this will be groupOfUniqueNames, but groupOfNames is a possibility, or you may have created a custom schema.
User ObjectClass Name	The name of the LDAP object class representing a user. inetOrgPerson is the standard schema, but you may have created a custom schema.
Member attribute name	<p>The name of the attribute on a group object that contains the names of members. If the group object class is groupOfUniqueNames, this will be uniqueMember. If groupOfNames, then member.</p> <p>The simple LDAP implementation assumes that group membership is represented by a multi-valued attribute on the group. If your directory server works differently, you will not be able to use Simple LDAP, and will need a custom repository class.</p>
Member group attribute name	The name of the attribute on a group object that contains the names of member groups. This is only relevant in cases where individual members and group members are added with different attributes, as in the case of the IBM Tivoli Directory Server. If this is omitted, it is assumed to be the same as the member attribute name.

Import disabled state	Whether or not to import the Swivel disabled state from the LDAP repository. If this is set to Yes, the following attribute is used to decide whether or not a user is disabled.
User disabled/enabled flag name	The name of the attribute on a user that indicates that the user is disabled or enabled. The inetOrgPerson schema does not allow for such an attribute, so to implement disabling users, you would need a custom schema. The use of two properties allows disabling to be handled in one of two ways: either a disabled flag is set to indicate that the user is disabled, or an enabled flag is set to indicate that the user is enabled.  If no such flag exists, the user is assumed to be active.
Ignore FQ name changes	Whether or not to treat an LDAP user object as the same user if the fully-qualified name changes, but the Swivel username remains the same.
Import Disabled/Ignore FQDN	Set the flags to import disabled state and ignore FQ name changes as described earlier.
Use SSL	Whether or not the server uses SSL for authentication

The server is now configured to pull in user information from the LDAP repository. The settings can be tested using the “browse in window” option, this should display the contents of the directory.

At this stage no groups within the repository are defined, so no users will be synched across. **4 Configure Groups.**

Go to Repository > Groups

Groups from LDAP repositories can be browsed and selected in the same way as described for Active Directory.

## Other repository types

Currently Swivel can pull user information from LDAP based user repositories. However the Swivel architecture allows for new repository types, e.g. SQL based, to be easily developed. Therefore if you have a requirement to pull information from a repository not covered by this manual contact [support@swivelsecure.com](mailto:support@swivelsecure.com)

## Deleting a Repository

A repository can be deleted by going to the Repository > Servers screen and clicking on the Delete button next to the appropriate repository. However, by default this only removes the repository definition from the current Swivel server, and not from the Swivel database. Neither does it remove users in that repository. To change this, on the Repository > Servers page, change the flag labelled “Delete users with server” to “Yes”.

The reason for including this option is to allow for the possibility of repositories being configured on more than one Swivel server.

If you unintentionally delete a repository with Delete users with server set to “No”, you can add the repository back in again (i.e. use the same name), change the flag, and delete it again.

## Integrating with Transport Classes

A transport class is the mechanism by which a security string is delivered to the end user for dual-factor authentication. The default method for this is via SMS and Swivel comes with a class that enable this via a GSM Modem. However the transport class can be used to interface Swivel with any suitable mechanism for carrying the string. Swivel comes with a number of pre-defined transports to choose from; others are available and new ones can be developed as required.

To use a transport class you need to configure destination attributes on Swivel. These represent a mapping of an attribute name to its definition within the connected repositories. For example to send security strings to a mobile phone an attribute called phone is required; this would map to phone for the local XML repository and telephoneNumber (or Mobile) for Active Directory.

### Transport>Attributes

Please enter the repository attributes for the transport types (e.g. Email, Mobile Phone).

Name:	<input type="text" value="phone"/>	
<b>Attribute:</b>		<input type="button" value="Delete"/>
local:	<input type="text" value="phone"/>	
activeDirectory:	<input type="text" value="telephoneNumber"/>	
<hr/>		
Name:	<input type="text" value="email"/>	
<b>Attribute:</b>		<input type="button" value="Delete"/>
local:	<input type="text" value="email"/>	
activeDirectory:	<input type="text" value="Mail"/>	
<hr/>		

Once the attributes have been defined they can be used as part of the transport definition; as shown in the table below.

Identifier	A name for the transport; this is especially important where multiple Swivel servers are sharing a database; see below
Class	The class that implements this interface
Strings per Message	For some transports you may wish to configure them to send multiple security strings within the same message. You can specify this here.
Destination Attribute	This is the attribute within the repository that will be used as the destination for the user. This will be one of the previously defined transport attributes.
Repository Group	The group in the repository that users will be a member of in order to use this transport to send their security stings.
Alert Repository	The group in the repository that users will be a



Group	member of in order to use this transport to send alert messages to.
-------	---

## Configuring transports in shared database Swivel installations

When a repository is synchronised with Swivel, the Swivel database stores the name of the transport that the user has been allocated. Where you have multiple Swivel servers connecting to the same database it is important that the transport configurations are consistent across the servers. The easiest approach is to ensure that the configuration is identical; however this is not the only approach and some extra resilience can be achieved.

For example, you can have two Swivel servers: one that uses a GSM modem to send SMS messages and another that is part of the same installation that uses an SMS provider. To achieve this, you would need to configure a transport called SMS on both servers and associate that transport with the same groups on both servers. However on one server the calls would be the GSM Modem class and on the other server it would be the class associated with the SMS provider.

**NB:** You can only use transports that use the same destination attribute.

**NB:** You need to configure transports on Swivel servers that are operating in Slave mode, even though they do not have a repository. The group definitions entered on the transport-general screen are not important but the transports still need to be configured.

## How to integrate Swivel with a GSM Modem

Swivel supplies a transport class that allows Swivel to send text messages via an AT compatible GSM modem. (The GSM modem obviously needs a valid SIM card and GSM coverage). The transport class operates the modem by sending AT Modem commands via the serial port. Therefore, before configuring the Swivel server, an AT Compatible modem needs to be connected to the serial port.

To configure the Swivel server to use the modem:

1. Go to the Transport > General tab.
2. Ensure that the modem class details are entered on the screen as shown below.
3. Enter a repository group name for the class, e.g. modem, if you are using the XML repository.
4. Click Apply.
5. Once this entry has been made, the transport will appear in the left hand pane under the Transport heading. You can then configure the modem interface as required from the Transport > GSM Modem screen.
6. To configure the modem interface select the serial port that you are going to use and set the other parameters to match those of the modem.
7. Configure the modem and connect it to the selected serial port.

### *Notes on Modems*

Swivel Secure can recommend a number of modems that we have tested with Swivel.

It is recommended that you configure the modem to a specific baud rate rather than using autobauding. You can effect this change by connecting to the modem via a terminal emulation program such as minicom (Linux) or Winterm (Windows) and typing the commands

```
AT+IPR=9600 (for 9600 baud)
```

```
AT&W To save the settings
```

The modem should respond with OK after each command

It is also recommend using hardware handshaking.

## How to integrate Swivel with an SMS service Provider

As an alternative to using a GSM Modem, Swivel can be configured to use an SMS service provider. To do this you need to obtain an account with an SMS provider that can deliver SMS messages to the Swivel user base. There are a number of such providers.

A transport class is required to act as an interface between Swivel and the SMS provider; it receives messages from Swivel and forwards them to the SMS provider in whatever format that the SMS provider requires.

Swivel has produced classes to work with a number of SMS providers including iTagg ([www.itagg.com](http://www.itagg.com)) and Clickatell ([www.clickatell.com](http://www.clickatell.com))

To configure Swivel to use an SMS service provider.

1. Obtain an account from an SMS provider for which there is a transport class available.
2. Go to the Transport > General tab.
3. Ensure that the SMS provider class details are entered on the screen as shown below.
4. Select a Repository group name for the class, eg smsUser. All members of this Repository Group will have their security strings sent to them via the SMS provider.
5. Select an Alert Repository group name for the class, eg smsUserAlert. All members of this Repository Group will have system alerts sent to them via the SMS provider.
6. Click Apply.

Identifier:	<input type="text" value="iTagg"/>
Class:	<input type="text" value="com.swiveltechnologies.pinsafe.transport.ITaggTransport"/>
Strings per message:	<input type="text" value="1"/>
Destination attribute:	<input type="text" value="phone"/>
Repository group:	<input type="text" value="smsuser"/>
Alert repository group:	<input type="text" value="smsAlert"/> <ul style="list-style-type: none"> <li>---NONE---</li> <li>PINsafeAdministrators</li> <li>PINsafeUsers</li> <li>Pinless</li> <li>pid</li> <li><b>smsAlert</b></li> <li>smsuser</li> <li>smtp2</li> </ul>

7. Once this entry has been made, the transport identifier (eg iTagg) will appear in the left hand pane under the Transport heading. You can then configure this interface as required from the Transport > iTagg screen.
8. In order to configure users to use this transport to receive their security strings, users need to be members of the appropriate repository group. If you are using the XML repository you can effect this by going to the User Administration screen, selecting a user, and selecting

smsUser as their transport group.

## Integrating via SMTP

Integrating with e-mail is a special case for integration. This is because Swivel requires integration with an e-mail server to send e-mail alarm messages. Therefore the actual integration details such as mail server IP address are specified in the Server -> SMTP screen. See SMTP (email) Loggin

## Integrating with Agents using Agent-XML

An agent is a piece of software residing outside the Swivel platform but which communicates with the Swivel platform to manage authentication. There are a number of ready made agents available, and new agents can generally be created very easily.

Agents communicate with Swivel via the Agent-XML interface.

This section covers how Swivel can be integrated with agents to support using Swivel with external applications such as websites. Any external application that needs to use the Admin API (new in Version 3.4) needs to be defined as an agent on Swivel.

### How to integrate Swivel with an external agent

The Swivel server will only service authentication requests from trusted agents.

The configuration of Swivel to work with an external agent therefore consists of adding the details of the agent to the Swivel trusted agent list. To do this go to the Agents screen, by clicking on Agents in the left hand navigation bar. Then enter a name for the agent, the IP address of the agent and then a shared secret.

- The IP address can be a single IP address or a range of IP addresses.

A range of IP addresses is specified using Classless Inter-Domain Routing (CIDR) notation, whereby you define an IP address but then specify how many bits actually are significant in terms of the network address. For example specifying 192.123.123.120 / 24 would allow authentication from any IP address in the range 192.123.123.xxx as only the first 24 bits of the IP address are required to match.

In order for authentication requests to be processed, the source IP address of the request and the shared secret presented by the agent need to match the details entered on this screen

### Server>Agents

Please enter the details for any PINsafe agents below. Agents are permitted to access the authentication services of the PINsafe server via the AgentXML interface.

Agents: Name:	<input type="text" value="jspsample"/>
Hostname/IP:	<input type="text" value="192.168.0.0/24"/>
Shared secret:	<input type="password" value="•••••"/>
Group:	<input type="text" value="---ANY---"/> <ul style="list-style-type: none"> <li>---ANY---</li> <li>PINsafeAdministrators</li> <li>PINsafeUsers</li> </ul>

It is possible to associate agents with groups within Swivel. When you configure an agent, if you leave the group selection as “ANY”, then any Swivel user will be able to authenticate via this agent.

If a group is specified then the user must be a member of that group in order to authenticate via this agent. The agent groups are similar to other groups in Swivel.

## Using the User ChangePIN application

One of the most common agents deployed with Swivel is the Change PIN application. This is an application that allows user to reset their own PINs. This application will come pre-configured as part of an appliance purchase.

This application can be deployed on the same server as the Swivel server or on a different server, but in either case it must be configured as an agent on the Swivel server.

The change pin application is usually supplied as a .war file. To install the application copy the changepin.war file to the webapps folder underneath the Apache Tomcat directory and then restart tomcat.

The settings for the Change PIN application are created by editing the settings.xml file under the webapps\changepin\WEB-INF directory.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
<entry key="ssl">>false</entry>
<entry key="server">localhost</entry>
<entry key="port">8080</entry>
<entry key="context">Swivel</entry>
<entry key="secret">secret</entry>
<entry key="redirect">http://www.google.com</entry>
</properties>
```

```
<entry key="ssl">>false</entry>
```

This sets whether to use ssl to connect to the Swivel server or not; default is not to use ssl.

```
<entry key="server">localhost</entry>
```

This tells the changepin application the server name or IP address of the Swivel server. The default setting is localhost; this appropriate for installations where the Change PIN application is installed on the same server as Swivel.

```
<entry key="port">8080</entry>
```

This is the port number being used by the Swivel server. The default is 8080; this will need to be changed if the Swivel server is using any other port, e.g. if it is using ssl for which the default port number is 8443.

```
<entry key="context">Swivel</entry>
```

This sets the context that Swivel servers are available on. The default is Swivel; generally the only time this would need to be changed is where there are multiple Swivel instances sharing the same Apache TOMCAT servlet container.

```
<entry key="secret">secret</entry>
```

This is the shared secret that needs to match the setting on the Swivel server for the change PIN agent. This should be changed to a suitably random setting.

```
<entry key="redirect">http://www.google.com</entry>
```

The redirect value is a url to where the user will be redirected once they have successfully changed their PIN; this is an optional setting.

You should now be able to use the Change PIN application. Go to <http://<ipaddress>:8080/changepin>

The Change PIN form will be displayed.

Enter a username and select Start Session; if the user is configured as a single channel user then a Turing image will be displayed.

Alternatively, if the user is a dual channel user, they may already have been sent a security string via SMS.

To change their PIN, the user must enter their one-time code based on their current PIN and then their new one-time code; therefore in the above example if the user has a PIN of 2378 that they wish to change to 9243 they would enter 0932 in the OTC box and 1089 the New OTC and confirm new OTC boxes.

To change their PIN, the user would then select change PIN.

**PIN change successful.  
Please wait while you are redirected. If your  
browser doesn't automatically redirect  
click [here](#) to continue.**

At which point they will be redirected to the defined redirect page if one was defined. If you check the Swivel logs you should also see the corresponding entries.

16:52:01 03 January 2007	INFO	192.168.0.13 changepin: Change PIN successful for user: test1.
16:51:47 03 January 2007	INFO	192.168.0.13 changepin: Session started for user: test1.

## How to integrate Swivel with an IIS Website

In this instance the Agent is implemented using an ISAPI Filter. This filter needs to be installed on the web server hosting the site to be protected. For information about filtering within IIS go to the Microsoft web site. <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/iissdk/html/22e3fbfb-1c31-41d7-9dc4-efa83f813521.asp>

To install the Swivel ISAPI filter you need to run the install SwivelIISFilter.exe on the IIS server. Make a note of the location the application is installed in (default is C:\Program Files\Swivel IIS Filter).

The configuration of the filter is achieved by running the configuration application, accessible from Start-All Programs - Swivel IIS Filter - Filter Configuration

The configurable Items are summarised below:

### Swivel Server

Hostname/IP:	The IP address or hostname of the Swivel server, which must be visible to IIS server
Port:	The port that the Swivel server services requests on; default 8080
Context:	The web application context in which the Swivel server is installed, usually Swivel
Secret:	The shared secret between this agent and the Swivel server. Needs to match value entered as part of Swivel config
SSL Enable:	Selected if SSL is being used between the agent and Swivel, otherwise false.
Permit Self-Signed:	Select if self-signed certificates are allowed.

### Authentication

Idle Time:	The connection idle time (in seconds) after which re-authentication will be required.
Username Header:	If the username needs to be included in the http header, what attribute will it be given
Channels:	What channels will be used, this will determine what buttons are presented on the authentication page, eg if you select Single Channel a get Image button will be included
Display Password Field:	This will include a password field if a password is being used as well as the PIN based authentication
Permit self-reset:	This will allow users to perform a self-reset via this agent.

### Exclusions

Addresses:	Comma separated list of IP addresses that are exempt from the requirement to perform Swivel authentication. Partial addresses such as "192.168" may be entered.
Paths:	Comma separated list of paths that are exempt from the requirement to perform Swivel authentication. Partial paths may be entered, for example "/images/" would allow access to "/images/logo.png" and "/images/staff.jpg"

### Inclusions

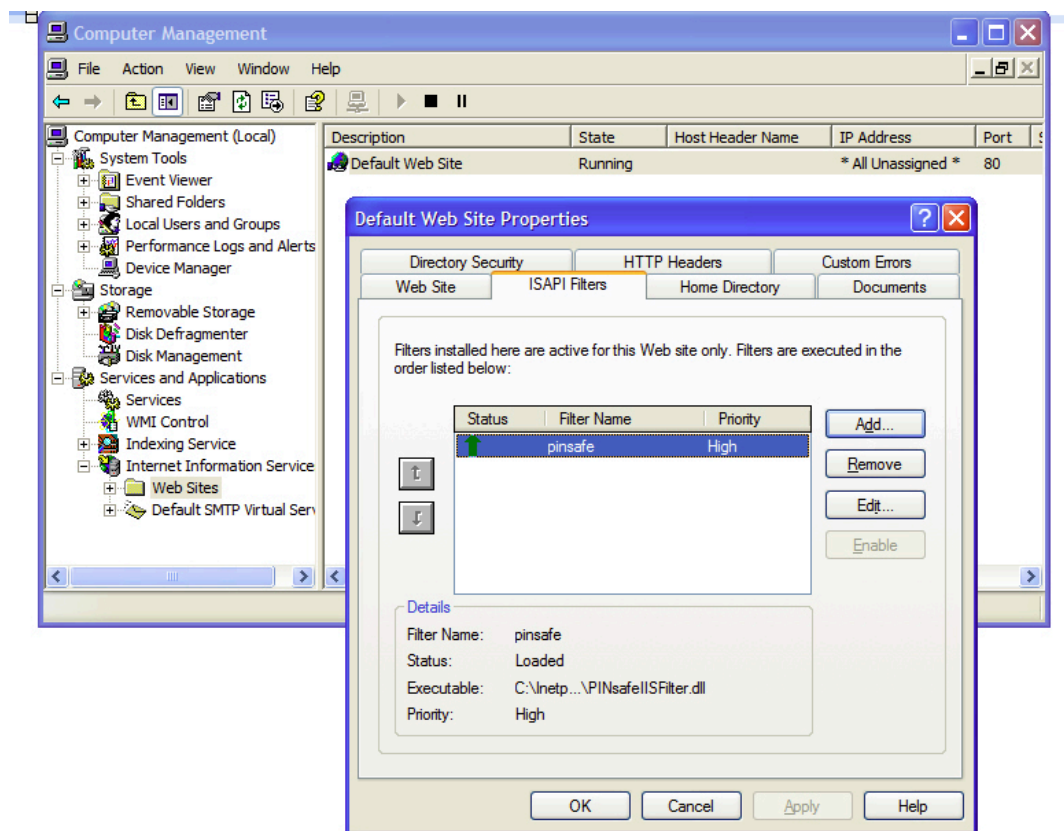
**Included Paths:** Comma separated list of paths that require Swivel authentication. If any paths are entered, then only those paths will be subject to Swivel authentication and any path exemptions will be ignored. Partial paths may be entered.

## Misc

**Default Path:** The default location to redirect a user to following login if no other destination is available. Setting this value is recommended as it will allow users who have navigated directly to the login page to access the protected resource

**Logout Path:** A path that causes the cookie to be deleted and authentication to be required on any subsequent request. This must be a path protected by the IIS filter

**Virtual web path:** The virtual web path for the IIS filter authentication pages



**Screenshot of IIS filter installation 1**

Once the filter has been configured, it can be applied to the web site.

In the Control Panel, select Administrative Tools, then Internet Information Services.

Navigate to the default web site (assuming that is the one you want the filter to apply to), right-click on it and select Properties.

Select the ISAPI filters tab, and click Add.

Enter the name "Swivel IIS filter" (you can call it what you like). Click the Browse button, and navigate to the installation folder noted earlier. Select SwivelIISFilter.dll and click OK.

Back in the main IIS management console, right-click again on the web site, select New, then Virtual

Directory.

When prompted for an alias, enter "Swivel". On the next screen, when prompted for a directory, click Browse and navigate to the "Web" directory underneath the filter installation directory. On the next screen, ensure that Read and Run scripts are enabled.

Once the virtual directory has been created, right-click on it and select Properties. On the Virtual Directory tab, click the Remove button next to Application name then click OK.

Once the filter has been applied to the website, when a user attempts to access part of the website protected by Swivel, Swivel will prompt them to authenticate. IIS may need to be restarted for filter settings to take effect.

## How to write an Agent

The Change PIN application and the IIS filter are two examples of Swivel agents that use the Swivel Agent-XML API. The Agent-XML API is a simple XML-based API that enables integrators to develop their own agents. This can provide for a very flexible authentication solution.

Swivel can provide a range of samples to help support this integration, for example how to make authentication requests from within a JSP. For more details e-mail [support@swivelsecure.com](mailto:support@swivelsecure.com)

## How to use the custom attribute

It is possible to assign a custom attribute to a user from the user admin screen. When a user successfully authenticates via the Agent-XML interface, this custom attribute is returned. This can be used to add granularity to the access rights granted on successful authentication.

When using the XML repository, this custom attribute can be set manually. When using Active Directory the custom attribute can be mapped to an Active Directory attribute.

## Integrating using RADIUS


Swivel can operate as a RADIUS server for working with external systems such as SSL VPNs. **NB: In order to authenticate via RADIUS, a user must be a member of the RADIUS users' group as defined on the Repository->Groups page.**

### *How to configure Swivel to operate as a RADIUS server*

In order to integrate Swivel with an external system the Swivel RADIUS server first needs to be enabled and then configured to receive authentication requests from the external systems, or Network Access Server (NAS) using RADIUS terminology.

The RADIUS server is configured by going to the Radius-> Server config page. Enter the required configuration details and then select Apply.



**RADIUS>Server** 

Please enter the details for the RADIUS server.

Server enabled:	<input type="text" value="Yes"/>
IP address:	<input type="text"/>
Authentication port:	<input type="text" value="1812"/>
Accounting port:	<input type="text" value="1813"/>
Maximum no. sessions:	<input type="text" value="50"/>
Permit empty attributes:	<input type="text" value="No"/>
Filter ID:	<input type="text" value="No"/>
Additional RADIUS logging:	<input type="text" value="Both"/>
Enable debug:	<input type="text" value="No"/>
Radius Groups:	<input type="text" value="Yes"/>
Radius Group Keyword:	<input type="text" value="POLICY"/>
<input type="button" value="Apply"/> <input type="button" value="Reset"/>	

**RADIUS server configuration page**

## Notes on RADIUS configuration

1. The IP address needs to be the 'internal' IP address of the Swivel server; this is the address to which the RADIUS server will bind. If you leave this setting blank the RADIUS server will process all inbound RADIUS requests received.
2. If you use different ports other than the default of 1812 and 1813, you will need to open up these ports on the appliance; see IPTables.
3. If you set Filter ID to YES the users username will be returned in the RADIUS Filter ID field of the RADIUS authentication response
4. Additional RADIUS logging will add RADIUS log events to the Swivel logs; therefore if a user authenticates via RADIUS there will be a log event for the successful RADIUS authentication and then another log entry for the Swivel authentication, as the example shows below.

12:13:08 03 January 2007	INFO	RADIUS: <158> Access-Accept(2) LEN=63 192.168.0.151:32859 Access-Request by greenford succeeded
12:13:08 03 January 2007	INFO	192.168.0.151 Aventura: Login successful for user: greenford.

## RADIUS Groups

Cisco, and other VPN vendors, use information passed back within a RADIUS Access-Accept response to determine what access policy to apply to the user.

Swivel can now support this form of operation for a number of vendors.

To implement this feature, set the RADIUS Groups setting to yes. The RADIUS group keyword can be used to filter the groups that are returned. Eg if you set the keyword to POLICY on group names that contain the word policy in their names will be returned.

This prevents irrelevant group information being returned. If this field is left blank then a list of all the groups of which the user is a member is returned. The format used to return the group list is set for each NAS.

On the NAS screen you select the relevant vendor from the Vendor (Groups) option.

Once the Radius Server is configured and enabled, it can be configured to accept requests from a NAS. This is achieved by selecting the Radius->NAS page and entering the details. These settings are the same as those for an agent, name, IP address, shared secret and group.

You can also specify that the NAS will use a secure RADIUS (EAP) protocol from the drop down list, the options are LEAP and EAP-MD5

## RADIUS>NAS

Please enter the details for any RADIUS network access servers. A NAS is authentication services of the PINsafe server via the RADIUS interface.

NAS: Identifier:

Hostname/IP:

Secret:

EAP protocol:

Group:

Authentication Mode:

Vendor (Groups):

Change PIN warning:

Two Stage Auth:

NAS configuration screen

For authentication requests to be processed by Swivel they need to come from the IP address specified in the NAS configuration, the shared secret needs to match that specified in the NAS configuration and be sent to the IP address specified in the Swivel Server Configuration.

## How to integrate a VPN with Swivel using RADIUS

In order to configure a VPN to use Swivel for authentication the VPN needs to be configured to make authentication requests against the Swivel server. This method for doing this depends on the VPN concerned so you need to consult the relevant documentation for the specific VPN. Swivel Secure will be able to provide you with specific instructions for a range of VPNs.

The basis of the configuration is basically to ensure that.

1. The VPN server needs to send authentication requests to the IP address and port number on which the Swivel Radius server is operating.
2. The VPN server needs to have its shared secret (and realm, if specified) set to match that on the NAS configuration screen.

This is all the configuration that is required for use with a VPN. If TURING image is being used and you require the TURING image to be integrated into the VPN log on page; the log-on page needs modification.

### *Two Stage RADIUS authentication*

One mode of operation for RADIUS integration is to use a two stage approach.

The user initially submits their username and password. If the password is correct Swivel responds with a RADIUS-Challenge which indicates to the VPN that additional credentials are required. If the user is a dual channel user and Swivel is set to on-demand authentication, the user will also be sent an SMS security string.

The VPN will then prompt for the additional credentials, in this case the one-time code. When the one-time code is entered it is submitted to Swivel via RADIUS. Swivel then checks the one-time code and if it is correct the user authenticates successfully.

This mode of operation has been successfully tested with Checkpoint, Cisco and Citrix products.

## How to use single channel with a VPN

In order to use single channel authentication with a VPN, the user needs to be able to obtain a TURING image, perform the OTC extraction using their PIN and then enter these details onto the VPN authentication page.

One way of delivering a security string to the user is to modify the VPN log-on page so that it incorporates a TURING image; like the example below. Some VPNs accommodate this form of integration; others require some customization and others make this impossible. Swivel have a number of these customizations available off the shelf.

However there are a number of ways of delivering the security string in a TURING image that do not require the VPN log-on page to be modified and thus are a very low-touch integration.

### *User Portal*

One solution to delivering the security strings is to use a user-portal web application to allow the user to obtain a TURing image via a web browser.

Swivel can supply a user-portal is can be hosted on any servlet container, including the server on which Swivel is running. It acts as an authentication agent for Swivel and can be used to deliver security strings to the end-user and also for functions such as allowing the user to change their PIN numbers. It is easily branded and customised.

The beauty of this solution is that it does not require and client to be installed on the user's laptop, all they need is a standard browser.

Users can bookmark this page and even have it on their desktop as a bookmark. Users can be sent the URL of the portal as part of the provisioning of Swivel.

**SWIVEL**  
AUTHENTICATION YOU CAN IDENTIFY WITH

- [Login](#)
- [Settings](#)

### PINsafe JSP Sample - Login

Username:

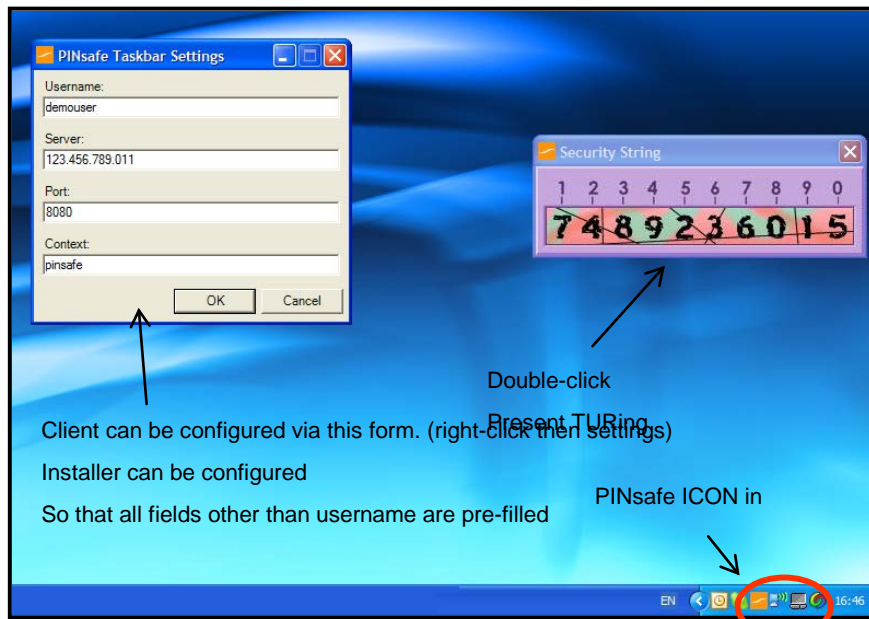
1	2	3	4	5	6	7	8	9	0
A	7	9	8	3	0	2	1	5	6

User Portal screen shot

To authenticate the user opens a browser at the portal's URL, enter their username (although this could be pre-filled, stored in a cookie). They select get image and the image is presented to them. The user performs the one-time code extraction and enters the one-time code into the VPN log-on form.

## SysTray Utility

Another way to deliver the TURing image to the end-user is to use the Swivel SysTray utility. This is a small client that resides in the users System Tray.



Screenshot of Systray Utility

The application can be manually configured to retrieve TURING images from any Swivel server, for any user. It can also be pre-configured so that the user need only enter their username. Once the application is configured, users can obtain a security string merely by double-clicking on the Swivel icon in the system tray (or by right-clicking and selecting get image)

The application is small and easy to distribute; it can be provided with its own windows installer.

The user of the systray application gives a true soft-token like experience.

These two options are not mutually exclusive: a single Swivel server can accommodate both alternatives.

## How to incorporate a TURING image into a VPN log on page

Example VPN screenshot

If you wish to integrate a TURING image with a VPN log on page then the VPN configuration needs modification.

The implementation of these modifications again depend on the VPN itself: some VPNs allow you to easily customize the log on page, for others it is more involved.

The log on page needs to

1. Have a “Start Session” or “TURING” button.
2. The user enters their username and then selects this button
3. This then fetches the Turing Image; this will be in the format

[http://Swivel\\_IP\\_Address:8080/Swivel/SCImage?username=demouser](http://Swivel_IP_Address:8080/Swivel/SCImage?username=demouser)

It is recommended that the request for the image is proxied via the VPN or via other means so that port 8080 on the Swivel server does not need to be opened up to the internet.

4. The user can then enter their OTC and select Sign In; this will then allow the VPN to use the username and extracted OTC to authenticate against the Swivel server.

## Integrating with 3<sup>rd</sup> Party Authentication systems

Swivel can be integrated with 3<sup>rd</sup> party authentication systems via its third party API. If this is required, a class needs to be developed that implements Swivel third party API, and interfaced to the third-party authentication system. The details of developing such as class are outside the scope of this document.

Assuming that this class exists, Swivel can be configured to use the third-party authentication system, via this class, for some or all users.

To configure Swivel to do this you need to go to the Sever>Third Party Authentication screen.

Enter an identifier for the Authentication, the class (that needs to be installed on the Swivel server), repository group that the user of the third party system will be associated with and any license key required by the third party system.

Once this configuration is in place the authentication process will be.

1. Agent submits authentication request, including credentials required by the third party system.
2. The Swivel server checks the user Swivel credentials.
3. If this stage of authentication is successful and the user is a member of the repository group associated with the third party authentication class, the Swivel server makes an authentication request to the third party system, passing the required credentials.
4. The third party class returns a success code and, if this stage of authentication is also successful, successfully authenticates the user.

## Integrating with other Authentication Servers

There are two ways in which Swivel servers can interact, enabling users from one server to authenticate to another: using peering, or by sharing a database.

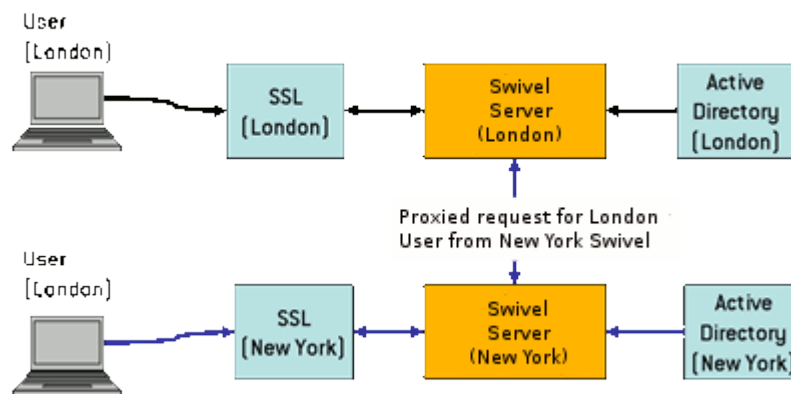
### *Peering Swivel servers*

It is possible to deploy a number of Swivel servers as a set of peers. Every user has an account on one of the Swivel peers, however but they can authenticate to any one of the Swivel servers.

For example, if a business has a London office and a New York office, running separate Active Directories and SSL VPNs a Swivel server can be installed in each office. Each Swivel server can be configured as a peer to the other.

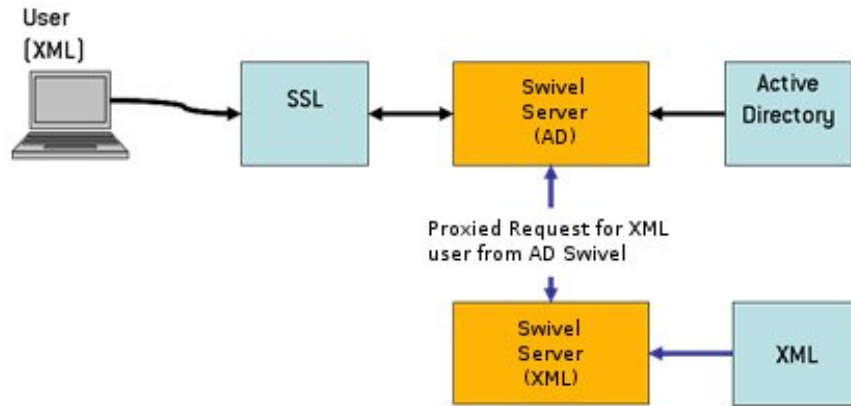
In this configuration, a London-based user can authenticate to the New York VPN; the New York Swivel detects that the user is served by the London server and proxies the authentication request to the London Swivel server.

The London Swivel server checks the user's credentials and returns the results to the New York server that, as a result, can allow or deny access.



**Example of peering for multi-office installation**

Another application of this is to support the inter-working of different user repositories, although this configuration is deprecated by Version 3.3's ability to connect to multiple repositories. One Swivel server can be configured to work with Active Directory, another to work with the XML repository. The two servers can be configured as peers so that a user from either user repository can authenticate to a VPN (or web application) authenticating via either Swivel server.



**Example of peering for multi-repository installation**

It is possible to deploy both these servers on the same Swivel appliance, thus avoiding increased hardware costs.

To implement peering, each Swivel Peer must know the IP address, context and port details of every other Swivel server in the peering network. Each Swivel server also has a shared secret used for authentication. To make authentication requests, the requesting server must present the shared secret to authenticate itself to its peer.

To set up this relationship, enter the details on the Server->Peers screen on each Swivel server. Using the London/New York example, on the London Swivel server the following would be entered.

**Server > Peers**

Please enter the details for any peer PINsafe servers below.

Peers: Name:

Hostname/IP:

HTTP port:

SSL:

Context:

RADIUS authorisation port:

RADIUS accounting port:

Shared secret:

**Example Peer configuration screen**

And then on the New York server would be entered



Peers: Name:	<input type="text" value="London"/>
Hostname/IP:	<input type="text" value="pinsafe.company.co.uk"/>
HTTP port:	<input type="text" value="8080"/>
SSL:	<input type="text" value="No"/>
Context:	<input type="text" value="pinsafe"/>
RADIUS authorisation port:	<input type="text" value="1812"/>
RADIUS accounting port:	<input type="text" value="1813"/>
Shared secret:	<input type="text" value="●●●●●●●●"/>

Example Peer configuration screen

Peering can be used for RADIUS and agent XML authentication solutions. An inbound RADIUS request will be proxied via RADIUS to a peer Swivel server as required. Similarly an Agent-XML based authentication request will be proxied via the Agent-XML interface.

Peering works by each peer keeping a record of the user-names active on the other servers. This list is updated periodically. How frequently and when this synchronisation takes place is configured on the server>jobs page: see section on Jobs. It is also possible to manual synchronise a peer from the user admin screen by pressing the Peer Sync button.

## Swivel as a RADIUS proxy

The RADIUS proxy option allows Swivel to work alongside other authentication servers to allow end-users to support a hybrid authentication model. This can be useful when users are being migrated from a token to a Swivel based solution.

### Server>Peers

Please enter the details for any peer PINsafe servers below.

Peers: Name:	<input type="text" value="test"/>
Hostname/IP:	<input type="text" value="192.168.0.10"/>
HTTP port:	<input type="text" value="8080"/>
SSL:	<input type="text" value="No"/>
Context:	<input type="text" value="pinsafe3.5"/>
RADIUS authentication port:	<input type="text" value="1812"/>
RADIUS accounting port:	<input type="text" value="1813"/>
Shared secret:	<input type="text" value="●●●●●●●●"/>
RADIUS Proxy:	<input type="text" value="Unknown User"/> <input type="button" value="Delete"/>

Setting Swivel as a RADIUS Proxy

Swivel can be configured to proxy to an external RADIUS server if the user is not a Swivel user or if the submitted credentials appear to be a Token Passcode.

This feature is configured on the Server->Peers screen.

## Multiple Swivel servers using a single database

From Swivel 3.2, it is possible for multiple Swivel servers to share a database. This was possible in 3.2, but it would require that the repository on each server was configured identically. With the introduction of multiple repositories, it is now possible for different Swivel servers to synchronise with different repositories, but for all servers to authenticate all users on all repositories.

In order to accomplish this, it is necessary to select an external database: the internal database is always local to the Swivel server, so cannot be shared between multiple instances. All that is required, therefore, is that the database URL is the same on all Swivel servers.

When repositories are synchronized into a shared database, the users can be authenticated by any Swivel server using that database, even if the user belongs to a repository that is not defined on that server. All repositories will be displayed in the User Administration screen, whether or not the repositories are defined on a particular Swivel server. The only difference is that users cannot be synced if the repository is not defined.

When connecting repositories to Swivel servers sharing a database, the repository name must be unique. If the same name is used for repositories on two different Swivel servers, it is assumed that they actually refer to the same repository. If this is not the case, it could cause users to be deleted every time the user sync runs. This is why the XML repository on each Swivel server is named from the server name, rather than simply being called "XML", for example.

One consideration here is that the username must be unique over all repositories within the database. It is not possible to have a user called, for example, "admin" in one repository, and another also called "admin" in a different repository, even if those repositories are synchronized on different servers.

## Operation & Maintenance

This section refers to Swivel running on a server, rather than as an appliance. For details of the operation and maintenance of appliances refer to the appliance How To guides.

### Tasks

#### *Log files:*

Tomcat creates a number of log-files under *usr/local/apache-tomcat-x.x-xx/logs*.

It is recommended that the older instances of these files are deleted to prevent them using up too much disk space.

#### *How to perform back-ups of the Swivel server*

To back up the user and configuration data for the server, the easiest approach is to back up the webapps folder under the Tomcat directory.

*usr/local/apache-tomcat-x.x-xx/webapps*

Where x.x-xx is the tomcat version number.

This will back up all the user and system data required by Swivel.

To ensure against sever hardware failure and to facilitate quick restore it is recommended that this back up should be stored on a remote server.

#### **Saving the configuration**

Swivel has a save configuration feature that allows the current configuration to be saved to an xml file. This provides a useful record of the Swivel configuration and can be useful for fault diagnosis

#### **How to perform automated backups**

The following guidelines are a suggested way of performing backups automatically in way that can be integrated with any existing enterprise back-up processes and systems.

This approach uses a back up script to copy the required files to a SAMBA share that allows the back-ups to be copied via the local network to remote server.

#### **Creating a Samba Share:**

This is a very quick guide to creating a public share using SAMBA, no security policies have been added to the share. It is being used as a form of access from your global backup server.

Firstly at the console, log in as root.

Start the x-display by typing *startx*

Once the graphical interface has started select *Applications - System Settings - Server Settings - Samba*

Select *Add*

In the Directory browse to and create a new folder called *SwivelBackup* in the */home/swivel* directory. Then select the new folder as the samba share directory.

N.B. The backup script provided later presumes a */home/swivel/SwivelBackup* folder exists, if you change this folder here remember to change the backup script to match

Select *Read* and *Allow access to everyone*

In the *Preferences - Server Settings - Security* menu item, set the Authentication Mode to *Share*

In the *Applications - System Settings - Server Settings - Services* menu item tick the *smb* service box and also select *Start Service*

You should now be able to access the share via a client machine pointing to [//<server\\_ip>/SwivelBackup](//<server_ip>/SwivelBackup) where <server\_ip> is the Swivel server's IP address.

### Writing a simple script to back up Swivel:

Log on to the console as root.

Use an editor to create a file `/home/swivel/backup.sh`

Enter the following code: **NB** replace the tomcat-x.x-xx version with the correct version number running on your server.

```
#!/bin/bash
/etc/init.d/tomcat stop
cd /usr/local/apache-tomcat-x.x-xx/webapps
tar --create \
--file=/home/swivel/SwivelBackup/Backup.tar \
--label= `Backup` \
--verbose \
Swivel
/etc/init.d/tomcat start
```

To make the script executable type:

```
chmod a+x /home/swivel/backup.sh
```

**Running this script temporarily halts the Swivel server and therefore prevents users authenticating via Swivel during back-ups.**

#### Running the script

To run the script manually, do the following:

```
/home/swivel/backup.sh
```

There should appear a new tar file called Backup.tar in your SAMBA share.

#### How often should I perform back-ups?

It is recommended that this script be run after the first user repository synchronization and then on a regular (e.g. weekly) basis. How regularly it is run depends on how often users and PINs change.

The script can be run as a cron job. The use of a SAMBA share means that this data can then be pulled from the machine and stored elsewhere to conform to any back-up routines/infrastructure that may already be in place.

These instructions are only a guideline and we would recommend you further your investigations into backup.

#### Restoring from Backup

If you should need to restore your server from a backup, carry out the following:

Log into the server as root and stop Tomcat:

```
/etc/init.d/tomcat stop
```

Delete the Swivel folder from within the tomcat webapps directory

```
cd /usr/local/apache-tomcat-x.x-xx/webapps
```

```
rm -rf Swivel
```

Copy and extract the backup file

```
cp /home/swivel/SwivelBackup/Backup.tar /usr/local/jakarta-tomcat-x.x-xx/webapps
```

```
tar -xvf /usr/local/jakarta-tomcat-x.x-xx/webapps
```

```
Backup.tar
```

Restart Tomcat and check Swivel is running and your settings are correct.

```
/etc/init.d/tomcat start
```

In a browser point to the admin console: [http://server\\_ip:8080/Swivel](http://server_ip:8080/Swivel).

### How to perform disaster recovery

In the event of a hardware failure or other scenario that requires a new server to be installed and brought up to the last recorded configuration of the live server.

1. Install Swivel and associated software from the disks supplied as part of the install.
2. Restore the latest back-up copy of the webapps folder to webapps
3. Restart the tomcat server

### Data Migration

The data migration feature allows you to move data from one database to another. For example of you want to change from using the internal database to an external database you can.

1. Prepare the new database
2. Add the details of the database to the Database – General screen.
3. Choose the Migrate option and select the new target database
4. Enter Migrate and click apply.

The data will be moved to the new database so you can use the new database without needing to re-provision users.

### Jobs

There are a number of processes (or jobs) that the server needs to run on a regular basis. These handle such things as synchronizing to the user repository and checking for any accounts that should be locked due to inactivity.

For the most part these settings can be left to their default values but there maybe reasons why an administrator would want to change these settings. When choosing these settings the administrator needs to balance the requirement to synchronize data regularly and the resultant loading on the server. Where possible these tasks should be scheduled to run during the server's quiet period.

### Session Clean Up

The session clean-up job is used to invalidate, after a given time, any security strings that have been requested by the user.

For example a session is deemed to have started when a TURING image is requested. The security string presented within that security string is only valid for as long as the session is valid. The length of time for which the session is valid is set by the session clean-up time: if it is set to 120 seconds the user will have 2 minutes in which to use the security string to authenticate. If they attempt to authenticate with that security string after that time their authentication will fail.

The same setting also applies to security strings delivered by SMS messages when they have been explicitly requested by the user, e.g. via a GET MESSAGE button. SMS messages sent automatically, after an authentication attempt, do not expire in this way.

The following jobs can be set using *cron* notation, this is described in Appendix E: Setting Schedules and CRON Strings

### ***Peer Synchronization***

This is similar to the user repository synchronization, only it refers to a server synchronizing its list of users with other peer servers within its peer network. Every time this job is run, the server will request an up to date user list from its peer servers.

### ***Inactive User Check***

If there are policies in place on the server to lock accounts that have been inactive for more than a certain time then this job will detect those inactive accounts and lock them.

### ***PIN expiry check***

If there are policies in place on the server that limit how long a PIN is valid for, this job will go through the user list and check to see the last time the PIN was changed and then either do nothing, send out a PIN expiry warning or lock out the account.

### **Audit Log tidy**

Swivel maintains an log of user activity for the users for a pre-determined period (as set on the Policy-General screen. The audit log tidy job deletes audit log entries that are now longer required to maintain this log.

## ***Logs and Alarms***

Swivel generates a range of log events and alarms they consist of the following options.

- XML Log files written to the local Swivel server
- Log files written to Syslog
- Alarm events sent as emails to a specified email address.

## **XML Logging**

XML logs are generated for a number of system events. They have varying levels of severity: FATAL, ERROR, WARNING and INFO. You can set the level of logging: setting the level to INFO will mean all events of severity INFO and above will be recorded.

## Logging > XML

Please specify how the server logs events to local XML files. These may be viewed or downloaded using the log viewer.

Level:	<input type="text" value="Info"/>
Filesize (KB):	<input type="text" value="256"/>
File count:	<input type="text" value="10"/>
Debug enabled:	<input type="text" value="No"/>
<input type="button" value="Apply"/> <input type="button" value="Reset"/>	

### Screen for configuring XML Logging

The log events are written to files on the Swivel server at <tomcat>\webapps\Swivel\WEB-INF\logs.

If you are running a backup script like the one described earlier in this manual then these log files can be included in that back up to provide a longer term log of system activity.

A log file is written to (as Swivel.log) until it reaches the file size specified on the Logging > XML screen. This file is then renamed to Swivel.log.1, a new Swivel.log is created and writing resumes to that file. This process repeats, creating Swivel.log.2, Swivel.log.3 etc.

The number of log files used is determined by the File Count entry. Once this count is reached, the oldest log file on the server is overwritten.

If debug is enabled, debug logs are created that give much more detailed information about the processes running within the server. This setting creates large log files and has an impact on the performance on the server and therefore should only be used for fault diagnosis. Debug logs are written to a separate file, <tomcat>\webapps\Swivel\WEB-INF\logs\debug.log

The contents of the XML log files can be viewed via the Swivel Administration interface, Log Viewer screen and can be downloaded from the Swivel server to a local machine.


## Syslog


As an alternative or addition to writing XML log files locally, Swivel can also write log files remotely by using the Syslog logging feature.

### Logging>Syslog


Please enter the details of an external syslog server to which PINsafe logging events should be delivered.


Syslogs: Host:

Level:  

Facility:  

Host:

Level:  

Facility:  

#### Syslog configuration screen

The logging level is set in the same way as for XML logging. The additional information required for Syslog is the host(s) to which the logs will be written and the syslog facility to be used.



## SMTP (email) Logging

Certain events can be emailed to operational managers. These are system errors, account lock-outs and account creations and deletions. The configuration screen for this feature is shown below.

To use SMTP logging you must have access to a suitable SMTP mail-server. The details for this server must be entered on to the Server > SMTP screen.


### Logging > SMTP

Please select which logging events are delivered as emails.

From:	<input type="text" value="PINsafe"/>
Send errors:	<input type="text" value="No"/>
Errors address:	<input type="text"/>
Errors subject:	<input type="text" value="PINsafe Error"/>
Send account locks:	<input type="text" value="No"/>
Account locks address:	<input type="text"/>
Account locks subject:	<input type="text" value="PINsafe Account Locke"/>
Send User Account Create/Delete:	<input type="text" value="No"/>
Account audit address:	<input type="text"/>
Account create subject:	<input type="text" value="PINsafe Account Creat"/>
Account delete subject:	<input type="text" value="PINsafe Account Delet"/>
<input type="button" value="Apply"/> <input type="button" value="Reset"/>	

SMTP Logging configuration screen

# Administrator's Guide

This section concentrates on explaining how the Swivel server is administered. It covers all the common tasks that an administrator would normally undertake. **NB** A full on-line help admin reference guide is provided as part of the Swivel distribution. Clicking the  icon will open a web browser with some help pages relating to that aspect of Swivel.

This section assumes all the required integration tasks described earlier in this document have been completed.

## Setting Policies

There are a number of policies relating to account/PIN management, for example PIN length and PIN validity periods. There are two modes of operation for Swivel: standard and PINless; these two modes are explained below:

### Standard

Users are assigned a PIN. They are sent a random security string and use their PIN to extract a one-time code.

### PINless

Users do not have a PIN, they are sent a random one-time code which they enter to authenticate.

Both modes can be supported on a Swivel server. Group memberships are used to define which mode a user uses. To make a group PINless, check the PINless property for the group in the Repository > Groups page.

#### Repository > Groups

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

	Single	Dual	Swivlet	Admin	Helpdesk	PINless	
Name: <input type="text" value="PINsafeUsers"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="Delete"/>
<b>Definitions:</b>							
ith000140: <input type="text" value="PINsafeUsers"/>							
LDAP: <input type="text" value="cn=PINsafeUsers,dc=test,dc=local"/>							
Name: <input type="text" value="PINsafeAdministrators"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="Delete"/>
<b>Definitions:</b>							
ith000140: <input type="text" value="PINsafeAdministrators"/>							
LDAP: <input type="text" value="cn=PINsafeAdmins,dc=test,dc=local"/>							
Name: <input type="text" value="PINlessUsers"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="button" value="Delete"/>
<b>Definitions:</b>							
ith000140: <input type="text" value="PINlessUsers"/>							
LDAP: <input type="text" value="cn=PINlessUsers,dc=test,dc=local"/>							
Name: <input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<b>Definitions:</b>							
ith000140: <input type="text"/>							
LDAP: <input type="text"/>							

Groups screen, showing PINless user group

Some policies apply to both PIN and PINless users, other are specific.

The general policies are set on the Policy > General screen.

## Policy > General

Please enter the policies to apply to authentication.

Security string type:	<input type="text" value="Numbers"/>
Auto. set credentials on user creation:	<input type="text" value="Yes"/>
Maximum login tries:	<input type="text" value="5"/>
Inactive account expiry (days):	<input type="text" value="4"/>
Non-Existent Users appear to be:	<input type="text" value="PINned"/>
Audit Log length (days):	<input type="text" value="30"/>

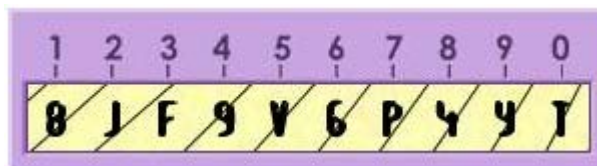
### General Policy Screen

#### ***Security String Type***

You can set the security string to be Numbers and/or Letters. You can have upper or lower case characters or a mixture of both. Mixed case is not recommended for TURING images as it can be difficult to differentiate between characters such as lower case l and number 1, even without the obfuscation.

You can even have a mixture of upper case letters and numbers. In order to make this option usable, letters that can easily be confused with numbers are not used. Therefore, if in doubt, a character is a number.

If the SMS delivery of security strings is used, then it is recommended that numbers are used for security strings.



Alpha-numeric security string; first character is an 8

#### ***Maximum login tries***

This is the maximum number of consecutive failed login attempts that a user can have before their account is locked out. See account unlocking accounts.

#### ***Inactive account expiry***

If an account is not used for a specified period, it will become locked. This setting allows you to specify that period (in days). If this is set to 0 then the user account never expires due to inactivity.

**Auto Credential setting**

If you set this parameter to YES, then whenever a new account is created, Swivel will automatically create a PIN for them. If a password is also required, this will also be generated. See Adding User Groups for more details.

**Non-Existent Users appear to be**

When a TURING image is requested for a user that does not exist, Swivel will still produce an image. This is to prevent a user determining which username are valid accounts. You can specify the type of image that is presented when a image for a non-existent user is requested.


Therefore if all your users have PINs you should set this to PINned. If all your users are PINless, this should be set to PINless and if you have a mixture of user you should set this value to mixed.

**Audit Log Length**

Swivel maintains a audit log of user activity; that can be extracted/interrogated by the Admin API. This setting dictates how long the activity records are maintained.

**PIN and OTC Policies**

PIN and OTC policies are set on the Policy > PIN and OTC screen. PINs within the Swivel product are the prime authentication credential and administrators may wish to replicate existing password management policies within Swivel.

**Policy>PIN and OTC** 

Please enter the policies to apply to PINs.

Minimum PIN size:	<input type="text" value="4"/>
PIN expiry (days):	<input type="text" value="0"/>
PIN expiry warning (days):	<input type="text" value="7"/>
Require PIN change after auto. setting:	<input type="text" value="No"/>
Require PIN change after admin. reset:	<input type="text" value="No"/>
PINless OTC length:	<input type="text" value="6"/>
Maximum repeated PIN digits:	<input type="text" value="0"/>
Allow numerical sequences for PIN:	<input type="text" value="Yes"/>
Auto-reset PIN on expiry:	<input type="text" value="Yes"/>
PIN change grace period (days):	<input type="text" value="0"/>
	<input type="button" value="Apply"/> <input type="button" value="Reset"/>

**PIN Policy Screen**

From this screen you can set the minimum PIN size (from 4 to 10 characters). You can also set a PIN expiry period. This determines how long a PIN is valid for: for example setting this figure to 90 days will ensure that users will need to change their PIN at least every 90 days. Setting this value to 0 (zero) would mean that the PINs would never expire.

You can ensure that users are warned about their impending PIN expiry. Setting the PIN expiry warning will determine how many days in advance the users will be prompted to change their PIN. The content and delivery of the warning will depend on which alert group the user is a member of.

Another PIN policy that Swivel can implement is to require a PIN change after the user has had a PIN auto-created (e.g. via auto credential creation) or where a user has had their PIN changed by the Administrator. Where these policies are in-force, a user must authenticate and then change their PIN: the PINs that have been created for them will only work once.

For PINless users, you can set a policy of how long the one-time code should be. This is set to 6 as a default, but can be from 4 to 8 digits.

For PINs you can also set policies on what PINs are valid. There are two optional policies that can be enforced:

- Maximum repeated digits. This indicates the maximum number of repeated digits that are allowed in the PIN. This allows the administrator to define what PINs are valid. E.g. it allows them to enforce a no repeated digits policy.
- Allow series. This sets whether ascending or descending arithmetic series are allowed as PINs. For example not allowing such series would prevent 1234, 2468, 6543 etc from being set as PIN numbers.

Administrators should be aware that enforcing these policies does reduce the number of possible valid PIN combinations and if they wish to enforce them all, they may wish to consider increasing the PIN length. These policies are enforced when users set their own PINs via the change PIN function and also when PINs are generated by Swivel. An administrator can override these policies and manually set any PIN number.

If you can specify the behaviour for when a PIN is due to expire. If you set Auto-reset PIN on expiry to Yes, a few days (set by the grace period) before a user's PIN expires, they are automatically sent a new PIN. If this is not set, the account will become locked.

## Password Policies

### Policy > Password

Please enter the policies to apply to passwords.

Require password:

Password mask:

Check Password with Repository:

#### Password Policy Screen

Passwords are an optional part of the Swivel authentication model. If required the Administrator can ensure that all accounts have a password as well as a PIN associated with them. If passwords are required then if auto-credential creation is set, Swivel will create a random password for the user. The random password will conform to the password mask. The password mask allows administrators to ensure that certain character types are included in the password in the specified order where: a = alpha, d = digit and s = special character. An example of a password conforming to the above password mask would be r4p&dl2a.

Another way of using passwords with Swivel is to use the password that the user has from their AD account (or other repository type). If the “Check Password with Repository” option is set to true, when a user attempts to authenticate no does Swivel check their OTC, but also checks the password they submitted against their repository. To do this the Swivel server must be able to connect to the user’s repository.

## User self-reset

### Policy > Self-Reset

Please enter the policies to apply to user self-reset.

Allow user self-reset:

Maximum self-reset tries:

#### Self Reset Policy Screen

Swivel supports a self-reset policy, whereby if a user’s account has been locked they can unlock it. They do this by being sent an unlock code via their alert transport; they then enter this code to

authenticate. The above screen enables this feature and stipulates the maximum number of reset tries a user is allowed.

## User Policies

Along with the server-wide policies the Administrator can set policies for individual users. To access this feature the Administrator can go to the User Administration screen and select a user and then select the Policy button. This will bring up the following screen for that user.

<b>Username:</b>	fred
<b>Created:</b>	16:07:13 20 April 2006
<b>Last login:</b>	16:08:51 20 April 2006
<b>Last PIN change:</b>	N/A
<b>Last self-reset:</b>	N/A
<b>Disabled:</b>	<input type="checkbox"/>
<b>Change PIN at first login:</b>	<input type="checkbox"/>
<b>PIN never expires:</b>	<input type="checkbox"/>
<input type="button" value="Reset"/>	<input type="button" value="OK"/> <input type="button" value="Apply"/> <input type="button" value="Cancel"/>

This screen shows the status of the user, when they were created on the system and when they last logged-in etc. The Administrator can also, from this screen, implement the following policies.

### *Disabled*

If an account is disabled a user cannot authenticate. Accounts can only be enabled again from this screen. Administrators may wish to disable accounts when a user no longer requires access but they wish to retain the information associated with that account. This feature is not available if the import disabled state on the repository->general screen has been set to Yes.

### *Change PIN at first login:*

The Administrator can ensure that the a user has to change their PIN at their next login; i.e. their PIN will only be valid for one authentication

### *PIN never expires*

The PIN for this account will never expire; this takes precedence over any server-wide PIN policy. Administrators may wish to use this feature for admin accounts.

## Single Channel

The Servers > Single Channel screen allows the Administrator to specify how single channel security strings are presented to the user. From this screen you can specify whether the security string image will be displayed as a TURing, PATtern or BUTTon image and whether the security string images will be animated. In this screen you can also determine whether to allow session creation via username. This means allowing a user to request a TURing image from any URL, for example by pasting

<http://<Swivel url>/Swivel/SCImage?username=<username>> into a web browser.

This provides flexibility in terms on security string delivery but does require that port 8080 is opened up on the Swivel server. See Protecting the Swivel Admin Console

## Dual Channel

The Server > Dual Channel screen determines how dual channel security strings are delivered. The default model for dual channel security strings is that a new security string is sent whenever a user attempts to authenticate, thus ensuring that the user always has a valid security string to use. An alternative is to only send a security string to the user when they explicitly request one; this is the on-demand authentication mode that can be enabled from this screen. If this mode is used the security string sent to the user is only valid for as long as the session time-out (clean-up) period specified on the server > jobs screen.

On-demand delivery means that a user can request a new security string at any time even when on-demand authentication is not enabled. This request is made via make a GET to `http://<Swivel url>/Swivel/DCMessage?username=<username>`

The request to send a dual channel security string can be via an agent configured on the Swivel server; alternatively Swivel can be configured to allow the request for a security string to be instigated from any IP address in a similar way to requesting TURING images. This is enabled by setting Allow message request by username to Yes.

You can also configure Swivel to return a confirmation image when a security is requested by setting Confirmation image on message request to Yes. This will return an image to indicate to the user whether the message request has been received successfully by Swivel.



**Message Request Confirmation Image**

This allows for a dual channel image to be requested in the same way as a TURING image from within JavaScript as well as providing useful feedback to the end-user. Note that the confirmation image confirms that Swivel has received the request not that the image has been sent.



## Alerting Users

The Transport > Alerts screen can be used to determine for which events users receive alerts. Alerts will be sent to the user based on which Alert Transport group to which they belong.

### Transport > User Alerts

Please select which alerts are delivered to users.

PIN expiry warning:	<input type="button" value="Yes"/> ▾
PIN change required:	<input type="button" value="Yes"/> ▾
PIN changed:	<input type="button" value="Yes"/> ▾
Account locked:	<input type="button" value="Yes"/> ▾
Device key allocated:	<input type="button" value="Yes"/> ▾
<input type="button" value="Apply"/> <input type="button" value="Reset"/>	

#### Alert Configuration Screen

## Managing Users

This section covers the management of users within the Swivel server; including the addition and removal of users, changing PINs and Passwords and unlocking accounts.

### Adding User Groups

The model for handling functionality within groups has been changed for Swivel 3.3, to accommodate the multiple repository facility. In earlier versions of Swivel, there were eight fixed groups on the Repository > Groups page, plus a group for each agent, transport class, third party authentication and RADIUS NAS. The group name was typed in, and the syntax had to be specific to the repository type.

In the new model, groups are all defined on the Repository > Groups page. Rather than being associated with a single function, each group can have as many functions as desired. Also, groups can be defined across multiple repositories. As many groups as are necessary can be defined. A new installation will have just two groups: normal users and administrators. However, as many groups can be defined as are necessary to give the desired access control.

To define a new group, all that is necessary is to enter a unique group name in the Name field in the blank section at the bottom of the Repository > Groups page. However, for that group to be useful, two additional pieces of information are required:

- What functionality the group is to have
- How the group maps to a group within each repository

Standard functionality is defined by setting the checkboxes for the group:

- **Single** - connect using single-channel authentication


- **Dual** – connect using dual-channel authentication
- **Telephony** – use PSTNsafe
- **Mobile Client** – use a mobile client
- **Admin** – administer the entire Swivel configuration
- **Helpdesk** – administer Swivel users
- **PINless** – user has no PIN, just a password

For users of previous versions of Swivel, these represent 6 of the 8 standard groups previously defined. There is no equivalent of the Swivel users group, as a member of any of the defined groups is by definition a Swivel user. The RADIUS group is also unnecessary, as in order to be a RADIUS user, a user must be a member of a NAS group.

To define additional functionality, a group must be selected on the appropriate agent, transport class, third party authentication or NAS. The groups on these forms are now drop-down boxes showing all defined groups, so you must add a group on the Repository > Groups page to correspond to the required functionality. For this reason, it is possible to define a group but not tick any of the boxes.

To associate a repository group with a Swivel group, you must enter something in the definition field for each repository. For the XML repository, simply copy the group name into the definition field. For Active Directory or LDAP, you will need to enter the fully-qualified domain name (FQDN) for the repository group. Leaving a definition field blank means that no users of that particular repository will be members of the appropriate group. There is no requirement to enter a definition for every repository in every group, although for the group to be useful, there must be a definition for at least one repository. Conversely, it is not necessary to define a separate group for each repository: if the functionality is the same for groups within different repositories, they can be defined as the same group within Swivel.

The Repository > Groups admin page will look something like the following:

**Repository>Groups** 

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

	Single	Dual	Swivlet	Admin	Helpdesk	PINless
Name: <input type="text" value="PINsafeUsers"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>Definitions:</b>	<input type="button" value="Delete"/>					
ith000140: <input type="text" value="PINsafeUsers"/>						
LDAP: <input type="text" value="cn=PINsafeUsers,dc=test,dc=local"/>						
Name: <input type="text" value="PINsafeAdministrators"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<b>Definitions:</b>	<input type="button" value="Delete"/>					
ith000140: <input type="text" value="PINsafeAdministrators"/>						
LDAP: <input type="text" value="cn=PINsafeAdmins,dc=test,dc=local"/>						
Name: <input type="text" value="PINlessUsers"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>Definitions:</b>	<input type="button" value="Delete"/>					
ith000140: <input type="text" value="PINlessUsers"/>						
LDAP: <input type="text" value="cn=PINlessUsers,dc=test,dc=local"/>						
Name: <input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>Definitions:</b>						
ith000140: <input type="text"/>						
LDAP: <input type="text"/>						

Repository Groups page

### Adding Users

To add users to Swivel you need to add them to an appropriate group within the repository and synchronise Swivel with that repository. If you have set Auto Create credentials to Yes, the user will be automatically sent a message with their account details in it. If Auto Create is not set you will need to manually reset or send out their PIN.

When adding users to the Swivel server you need to consider the following:-


- What rights will they have, e.g. will they be able to use single channel, dual channel or both
- How will security strings be delivered to them
- How will their PIN (and Password) be created and delivered to them.

The implementation of these will be achieved by ensuring that the user is a member of an appropriate user group. See the previous section for details.

### Allocating users to authentication methods

If the user accounts are being synchronized with Active Directory, the user must be a member of the correct group(s) within Active Directory to be able to use the associated authentication methods within Swivel. The Repository > Groups screen will show of which Active Directory groups the user must be a member. Different authentication options may be associated with different groups within Active Directory or, as in the example below, all users will have access to all authentication types.

---

**Repository>Groups** 

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

	Single	Dual	Swivlet	Admin	Helpdesk	PINless	
Name: <input type="text" value="PINsafeUsers"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="Delete"/>
<b>Definitions:</b>							
ith000140: <input type="text" value="PINsafeUsers"/>							
LDAP: <input type="text" value="cn=vpnsslusers,ou=group,dc=banquecramer,dc=ch"/>							
Name: <input type="text" value="PINsafeAdministrators"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="Delete"/>
<b>Definitions:</b>							
ith000140: <input type="text" value="PINsafeAdministrators"/>							
LDAP: <input type="text" value="cn=vpnssladmins,ou=group,dc=banquecramer,dc=ch"/>							
Name: <input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<b>Definitions:</b>							
ith000140: <input type="text"/>							
LDAP: <input type="text"/>							

### Simple Active Directory Group Structure

In addition you can specify that one or more groups for users will operate “PINlessly”. To implement this, simply make sure that the PINless box is checked for the appropriate group(s).

***Allocating Users to Transports***

Users need to be associated with transport classes for the delivery of security strings and for system alerts, e.g. notification of PIN.

The Transport > General screen shows which groups are associated with which transports. It also shows the destination attribute; therefore the Administrator needs to ensure that this attribute is set for the user.

Identifier:	<input type="text" value="GSM Modem"/>
Class:	<input type="text" value="com.swiveltechnologies.pinsafe.transport.GsmTran"/>
Strings per message:	<input type="text" value="1"/>
Destination attribute:	<input type="text" value="phone"/>
Repository group:	<input type="text" value="GSMModemUsers"/>
Alert repository group:	<input type="text" value="GSMModemUsers"/>

**Screen showing AD groups associated with GSM Modem transport.**

The Administrator may wish to use different transports for alerts from security string delivery.

***Allocating Users to Agents***

It is possible to configure Swivel so that only certain users can authenticate via certain agents. When an agent is added to Swivel it may have had a group associated with it. If this is the case then the user must be placed in that group in order for that user to be able to authenticate via that agent. If no group is specified, any user can authenticate using that agent.

***Creating the account***

Once the user has been made a member of the relevant groups the repository can be synchronised with Swivel; this can either be done manually via the User Admin screen or automatically be setting up a job to perform this. (Server-Jobs, see Operation & Maintenance)

Swivel will create a new account; if auto-credentials create is enabled, Swivel will also create a username (and password) if required and send them to the user via their allocated Alert transport.

### Unlocking Accounts

Accounts can be locked or disabled. If a user's account is locked or disabled they will not be able to authenticate via Swivel.

You can configure Swivel to send an email to an administrator to inform them when an account has become locked. See Logs and Alarms.

The main status screen on Swivel immediately indicates if there are locked or disabled accounts on the Swivel server.

## PINsafe Status

<b>Licensed to</b>	
<b>Licensed users</b>	5
<b>User accounts</b>	3
<b><u>Locked user accounts</u></b>	1
<b><u>Disabled user accounts</u></b>	1
<b>Active user repository</b>	XML
<b>RADIUS server enabled</b>	No
<b>Server IP address</b>	127.0.0.1
<b>Server hostname</b>	ITH000136

Swivel Status Screen

The entries for the Locked and Disabled are hyperlinks to the user administration page; clicking on these links will take you to a list of all Locked accounts and all Disabled accounts respectively.

On the User Administration screen, locked accounts are shown in **bold** and disabled accounts are in *italics*.

test10	▼	✓	✓	✓	✓	✓	
<b>test11</b>	▼	✓	✓	✓	✓	✓	
<i>test12</i>	▼	✓	✓	✓	✓	✓	
test13	▼	✓	✓	✓	✓	✓	
test14	▲	✓	✓	✓	✓	✓	
<input type="button" value="Edit"/> <input type="button" value="Policy"/> <input type="button" value="Reset PIN"/> <input type="button" value="Reset Pwd"/> <input type="button" value="Resend"/> <input type="button" value="Unlock"/> <input type="button" value="Delete"/>							
test15	▼	✓	✓	✓	✓	✓	

Accounts listed on User Administration screen

To unlock an account, click on the account in question and then on Unlock. To enable a disabled account, click on the Policy button and uncheck the disabled checkbox.

**Viewing and Sending Security Strings**

Username	Admin	Helpdesk	Single	Dual	Swivlet	PINless
admin	✓	✓	✓	✓	✓	
helpdesk		✓	✓	✓	✓	
test	✓	✓	✓	✓	✓	

**Accounts listed showing Send String option**

The user can be sent a new security string from the admin console by selecting the user and clicking the Send String button. Selecting the View Strings option allows a helpdesk user to view the security string that has been sent to the user.

**Username:** test

**Single Channel**

1	2	3	4	5	6	7	8	9	0
2	5	9	0	3	1	8	4	6	7

**Dual Channel**

1	2	3	4	5	6	7	8	9	0
7	2	1	4	8	6	5	3	0	9

**View Strings screen**

Selecting Show on the single channel option will create and display a new single channel string. Selecting Refresh on the Dual Channel option will generate a new dual channel string for the user and instigate the sending of this security string to the user.

**Resetting/Resending Credentials**

You can manually set a PIN to a know value if required by selecting the Reset PIN option. Alternatively you can make Swivel create a new PIN for a user and set that new PIN to the user via their Alert Transport if they have one configured.

This is useful where a user has forgotten their PIN or lost the credentials alert message. It is recommended that the logs are checked to ensure that the new PIN message has been successfully despatched.

## Appendix A: Building and Deploying Swivel server

### Upgrading from Swivel version 3.2 onwards

These instructions refer to standalone Swivel servers only, separate instructions apply for upgrading HA appliances, contact support@swivelsecure.com for more details.

Upgrading from version 3.2/3.3 should be transparent. Both the user database and the configuration will be upgraded automatically. To upgrade from an existing installation of Swivel 3.2,3.3 and 3.4 follow the following procedure:

1. Stop the Tomcat Server (service tomcat5 stop).
2. It is **highly recommended** that you take a copy of the entire ...\\Tomcat x.x\\webapps\\Swivel\\WEB-INF\\ folder but specifically ensure steps 3,4,5 and 6 are completed
3. Back up the Swivel configuration by taking a copy of
4. ...\\Tomcat x.x\\webapps\\Swivel\\WEB-INF\\conf\\config.xml to a safe location
5. Back up the Swivel User Repository by taking a copy of ...\\Tomcat x.x\\webapps\\Swivel\\WEB-INF\\data\\repository.xml to a safe location
6. Back up the Swivel user data by taking a copy everything under ...\\Tomcat x.x\\webapps\\Swivel\\WEB-INF\\db to a safe location
7. Back up any customized transport, user repository classes residing on the Swivel server.
8. Start the Tomcat Server (service tomcat5 start)
9. Remove the Swivel.war file from webapps, ensuring that you have a back-up copy. This should undeploy Swivel and the Swivel folder should be deleted
10. Stop Tomcat. If the webapps\\Swivel folder has not been deleted manually delete it.
11. Copy the new Swivel.war file to the webapps folder
12. Start the Tomcat server(service tomcat5 start): this will deploy the Swivel application.
13. Stop the server
14. Copy the backed-up config.xml file from step 3 to ...\\Tomcat x.x\\webapps\\Swivel\\WEB-INF\\conf, overwriting the installed version.
15. If you are using an XML repository, restore the backed-up copy from step 4 to ...\\Tomcat x.x\\webapps\\Swivel\\WEB-INF\\data, overwriting the installed version.
16. If you are using an internal database, restore the database files backed up in step 5 to ...\\Tomcat x.x\\webapps\\Swivel\\WEB-INF\\db.
17. Ensure that the ownerships and permissions on the file that you have restored are correct

```
drwxrwxr-x  2 swivel swivel  4096 Dec  8 12:13 db
```

For example to change the ownership of data run the chown command from the WEB-INF directory.

```
chown -R swivel:swivel db
```

To change the permission of data run the chmod command from the WEB-INF directory

```
chmod -R 775 db
```

18. Restart the server.
19. After upgrading you will see that any repository you had will exist on the new installation and it will be named after the associated hostname (or hostname followed by context in the case of the XML repository). Therefore if you had an AD repository in 3.2 you will have an XML repository in 3.3 named after the AD server's hostname (or IP address)
20. If you had an AD repository on your 3.2 installation you will now have the option to add an XML repository. It is recommended that you do this as this gives you the ability to create admin accounts whilst you are working on the installation

## Upgrading from Swivel version 3.1

To upgrade from Swivel version 3.1, follow the same procedure as above, except that you should NOT restore config.xml in step 11. Configurations from versions earlier than 3.2 cannot be upgraded automatically. Instead, you should follow the following steps after completing step 15 above.

1. Log into the administration console using the username “admin” and the PIN 1234.
2. You will now have the new version (running under /Swivel) running alongside the existing version (now running under /Swivelold). You can then cut and paste configuration settings from one to the other.
3. **NB:** You may want to delete the user sync job whilst performing the configuration as if the repository synchronizes prior to the repository groups being configured then user accounts may be deleted. To do this, go to the Repository section and select the last sub-entry: this should be labelled with the Swivel server computer name. Delete the Synchronization Schedule entry and click Apply.

## Architecture

The Swivel application runs within a JSP/Servlet Container, Apache Tomcat being the default that is recommended, specifically version 5.5.23. Swivel has also had some testing under later versions of Tomcat as well as Jetty and Glassfish with no problems observed; however, these containers are not yet recommended for use with Swivel, and earlier versions of Tomcat (prior to version 5) or Jetty are known not to be compatible.

Swivel also requires a Java JRE. Specifically, Java JRE 1.6 is recommended, and is the minimum requirement. Certain configurations of the product the Java Communication API needs to be installed: more details in the installation section.

Swivel needs a database server to store user data and credentials. An internal database server is provided, but there is also the option to use an external database server. The list of currently supported database servers can be found in the Database > General section of the administration console. Note that in order to use any of these databases you will need the appropriate JDBC driver. These are not provided with Swivel because of licensing regulations, but can usually be obtained free of charge from the database manufacturer.

Swivel can use its own repository for sourcing user accounts, or can be configured to work with an existing Active Directory or other LDAP-based user repository: see “How to guides” in the Appendix.

Note: An important part of this document is the record of the installation, covered in Appendix D: Swivel Installation details. It is recommended that this section is completed at the time of installation and kept with the server or elsewhere where it can be found easily if required. A copy of the installation record needs to be sent to Swivel Secure for their records; this can be sent to:

e-mail [support@swivelsecure.com](mailto:support@swivelsecure.com)  
 fax +44 1937 547 590

Swivel can be purchased as an Appliance or software only. If an Appliance is shipped with Swivel, all software will come pre-loaded, there is an Appliance User guide that covers the appliance specific configuration issues such as changing IP addresses etc.

Please note that the installation should only be attempted by someone comfortable with this kind of installation. Also if you install Swivel on your own hardware platform and OS that you should take all the necessary steps to security harden the installation.

Before you deploy and start Swivel, ensure you have the following software installed:

- Java runtime environment (JRE) Version 1.6, available from [www.sun.java.com](http://www.sun.java.com)
- Tomcat v5.5 available from <http://tomcat.apache.org/> Version 5.5.23 recommended.
- Java Communications API if you intend using a GSM MODEM (follow the Java Comm API installation instructions) See [Appendix A](#) for instructions. This is available from



<http://java.sun.com/products/javacomm/downloads/index.html>

- The Swivel distribution (.war) file

If you have an appliance the default Linux password is lockbox. This applies for root and the user swivel. These passwords should be changed using the linux passwd command.

## Swivel Deployment

This section refers to a “self-install”, an Appliance comes with the software pre-installed. To deploy and start Swivel perform the following steps:

1. Place the war file in Tomcat's webapps folder
2. Start Tomcat
3. Browse to [http://Swivel\\_server\\_IP\\_address:portnumber/Swivel](http://Swivel_server_IP_address:portnumber/Swivel) (<http://localhost:8080/Swivel> for example)
4. The default administrator is 'admin' with a PIN of '1234' so enter admin in the username click start session, then enter the first 4 digits of the security string. Click login.

## Protecting the Swivel Admin Console

In order for users to retrieve TURING images from Swivel, the Swivel server needs to be accessible via the internet; this requires the opening up of port 8080 by default.

As the Swivel admin console is also on port 8080, measures need to be taken to protect the admin pages from unauthorised access. Frequently a customer will have a proxy that they can configure to proxy the inbound image requests. Where this is not available (and even where it is) the Admin-Console filter can ensure that access to the Swivel Admin Console is only available from a predefined set of IP addresses.

The Swivel Admin-Console Filter is an implementation of a [J2EE Servlet Filter](#) that is deployed against the Admin-Login servlet.

## Configuring the filter

The filter configuration is controlled by two files found in the WEB-INF folder:

filter.properties	Determines the way the filter behaves when access is denied or granted
ranges.xml	List of IP ranges that can access the Admin Console

These files are read as TOMCAT initializes the filter; therefore changes to these files will only take affect after TOMCAT has been restarted.

## Editing filter.properties

The default filter.properties file is shown below.

```
#
# Admin Console Filter Localization
#
# Commented lines will result in no message being logged
#
# ALLOWED = Access Allowed
DENIED = Access Denied
ERROR = Page Not Found
```

```
# FILTERING = Filtering
STATUS = 404
```

The entries are as follows:

Entry	Meaning	Default
ALLOWED	Message written to TOMCAT console with request IP address when the filter allows access.	Commented out; filter is silent
DENIED	Message written to TOMCAT console with request IP address when the filter denies access.	Access Denied
ERROR	Message reported back to browser when access is denied. If not set, no response is sent and the browser will eventually time out.	Page Not Found
FILTERING	Message written to TOMCAT console followed by address ranges as TOMCAT initializes the filter.	Commented out; filter is silent
STATUS	The http status code reported back when access is denied. This should match the error message.	404

## Editing ranges.xml

The ranges.xml file holds the list of IP addresses that are allowed to access the admin console. The default ranges.xml file is shown below.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE properties (View Source for full doctype...)>
<properties version="1.0">
<entry key="anyone">0/0</entry>
<entry key="localhost">127.0.0.1/255.255.255.255</entry>
</properties>
```

The default configured ranges are named “anyone” and “localhost” and represent access from any IP address and localhost only respectively.

An address range is specified as an IP address followed (optionally) by a mask. The mask can be a single integer representing the number of significant address bits that must match for access to be allowed or it can be an IP-style dotted decimal. Both styles are present in the default file, but further examples are shown below.

IP Range	Meaning
0/0	A /0 mask means that no bits need to match in the address. This allows access from all IP addresses.
123.123.123.123/0	
127.0.0.1/32	A /32 mask means all 32 bits must match.
127.0.0.1/255.255.255.255	The equivalent dotted-decimal is 255.255.255.255. Specifying no mask is the same as specifying a /32 mask.
127.0.0.1	

192.168.0.0/24  
192.168.0.0/255.255.255.0

Access will be allowed from any address on  
the 192.168.0 subnet.

The default entries allow access from all IP addresses. Removing the entry for “anyone” will restrict access to localhost. Further ranges can be added to ease administration. All ranges should have a unique name.

## Activating the Filter

The filter is applied to the admin console by specifying the filter-servlet relationship in the web.xml file found in the webapps/Swivel/WEB-INF folder. In versions of Swivel that have the filter bundled there is no need to change this file. If the filter is being used with a 3.1.x version of Swivel then the web.xml file will need editing.

The segment that needs inserting is in the web.xml.fragment file and is shown below.

```
<filter>
  <filter-name>AdminConsoleFilter</filter-name>
  <filter-class>com.swiveltechnologies.filter.AdminConsoleFilter</filter-class>
</filter>
<filter-mapping>
  <filter-name>AdminConsoleFilter</filter-name>
  <servlet-name>AdminLogin</servlet-name>
</filter-mapping>
```

The <filter> element describes the filter and the <filter-mapping> element applies it to the AdminLogin Servlet and the fragment needs inserting at the top of the web.xml file, before the first servlet definition, as illustrated below.

```
<filter>
  <filter-name>AdminConsoleFilter</filter-name>
  <filter-class>com.swiveltechnologies.filter.AdminConsoleFilter</filter-class>
</filter>
<filter-mapping>
  <filter-name>AdminConsoleFilter</filter-name>
  <servlet-name>AdminLogin</servlet-name>
</filter-mapping>
<servlet>
  <servlet-name>QuartzInitializer</servlet-name>
</servlet>
```

To activate these new settings TOMCAT must be restarted.

## Filter in operation

When someone attempts to access any part of the admin console they are redirected to the admin log-in page. At this point the filter intercepts the request and checks to see if the IP address is on the allowed list. If it is not it returns the error code and message defined in the filter.properties file.

## Session Sharing

Swivel will use multicast for session sharing, therefore no IP addresses need specifying. Swivel will use session sharing if there is a cache.xml defined under the webapps/Swivel/WEB-INF/classes. It is recommended that anyone wishing to use session sharing consult with a Swivel Technician for advice as there are certain requirements on the network infrastructure.

## Appendix B: Windows Installation of the Java Comm API

1. unzip the file javacomm20-win32.zip into the root of C:

This will produce a hierarchy with a top level directory commapi.

This example assumes that you have installed the J2SE version 6 and has been installed into C:\Program Files\Java\jdk1.6.0\_06

2. Copy win32com.dll to your Java\jdk...\jre\bin directory.

```
C:\>copy c:\commapi\win32com.dll to C:\Program Files\Java\jdk1.6.0_06\jre\bin
```

3. Copy comm.jar to your Java\jdk...\jre\lib\ext directory.

```
C:\>copy c:\commapi\comm.jar to C:\Program Files\Java\jdk1.6.0_06\jre\lib\ext
```

4. Copy javax.comm.properties to your Java\jdk...\jre\lib directory.

```
C:\>copy c:\commapi\javax.comm.properties to C:\Program Files\Java\jdk1.6.0_06\jre\lib
```

5. Go to Environment Variable and add the Java\jdk...\bin to 'path'

```
C:\Program Files\Java\jdk1.6.0_06\bin
```

### Testing

The Comm API also comes with a number of samples that can be used to confirm the Comm API has been installed correctly.

One of these samples is called BlackBox.

To use Blackbox add BlackBox.jar to the CLASSPATH in Environment Variables

```
C:\commapi\samples\BlackBox\BlackBox.jar
```

To run BlackBox, open a command prompt and go to C:\commapi\samples\BlackBox. Then enter 'java BlackBox'

```
C:\commapi\samples\BlackBox>java BlackBox
```

## Appendix C: Active Directory/LDAP Groups and Attributes

Using Active Directory groups to specify how users are managed is a quick and flexible way to manage Swivel users. You can create a set of groups in such a way that different user groups have a range of different user experiences and rights within Swivel, or if you have a very simple Swivel installation you can use just two or three different groups to very simply manage your Swivel users. Note that although this section refers consistently to Active Directory, all comments apply also to repositories configured as Simple LDAP. Note also that the groups defined within Swivel (as of version 3.3) can refer to a single group within each of the defined repositories, or can refer to a single repository only.

The considerations that should be taken into account when planning integration with Active Directory are:

What authentication modes are to be used and whether all users are going to be able to use all methods

- Dual Channel
- Single Channel
- A mixture of the two

What delivery method is to be used for alerts (e.g. account creation) and dual channel authentication, and are all users going to use the same method.

How is the user PIN to be created?

Have all the Active Directory accounts been fully populated with information, e.g. Mobile phone numbers (without +, e.g. 4412345678), Email addresses

Plan your Active Directory Groups. Ideally, by using a hierarchy of groups, users should be able to be assigned a member to one group to give all the requirements for Swivel.

### User rights

The following standard rights are defined on the Swivel server. Each group can be assigned any combination of these rights.

Single	Members of groups with this right can authenticate via the single channel (TURING image) interface.
Dual	Members of groups with this right can authenticate using dual channel authentication, e.g. using SMS messages
Telephony	Member of groups with this right can authenticate using the PSTNsafe server.
Mobile Client	Members of groups with this right can authenticate using the Swivel mobile client.
Admin	Members of groups with this right can configure Swivel
Helpdesk	Members of groups with this right can manage users, e.g. reset PINs, passwords etc. Users with the Admin right automatically also have this right.
PINless	Members of groups with this right will be PINless. They will use one-time passcodes rather than security strings and one-time codes.

In addition groups will need to be defined for the transports that users are going to use, the agents via which the users will be allowed to authenticate, which RADIUS NAS can be used to authenticate, and the use of third party authentication methods.

### Simple AD Group and Attribute example

A simple Swivel installation may be where TURING image authentication is being used to authenticate a VPN. All users will be single channel users. All users will be sent alerts via e-mail. All users will authenticate via the same NAS. The same group of users will configure and operate Swivel

so there is no requirement for separate helpdesk/admin roles. All users will be PINless.

These requirements can be met by creating a simple set of two groups.

Group name	Example Active Directory definition
SwivelUsers	CN=SwivelUsers, OU=Swivel, DC=example, DC=com
SwivelAdministrators	CN=SwivelAdministrators, OU=Swivel, DC=example, DC=com

The SwivelUsers group will be given the Single right, allowing them to authenticate using single channel. They will also be given the PINless right. The SwivelAdministrators group will be given the Admin right. Since this also includes HelpDesk rights, there is no need to tick both.

Authentication for the SwivelAdministrators group can be handled in one of two ways: either the SwivelAdministrators group is itself a member of the Swivel users group within Active Directory, in which case they automatically get the Single and PINless rights, or else the group can be explicitly given these rights.

The Repository -> Groups screen will look as shown below (this example assumes the administrators group is a sub-group of the users group):

	Single	Dual	Swivlet	Admin	Helpdesk	PINless
Name: <input type="text" value="PINsafeUsers"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>Definitions:</b>	<input type="button" value="Delete"/>					
ith000140: <input type="text" value="PINsafeUsers"/>						
AD: <input type="text" value="CN=PINsafeUsers, OU=pinsafe, DC=example, DC=c"/>						
Name: <input type="text" value="PINsafeAdministrators"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>Definitions:</b>	<input type="button" value="Delete"/>					
ith000140: <input type="text" value="PINsafeAdministrators"/>						
AD: <input type="text" value="CN=PINsafeAdministrators, OU=pinsafe, DC=examp"/>						

On the Transport->General screen the email (SMTP) transport will be configured as follows:

Identifier:	<input type="text" value="SMTP"/>
Class:	<input type="text" value="com.swiveltechnologies.pinsafe.transport.SmtpTra"/>
Strings per message:	<input type="text" value="1"/>
Destination attribute:	<input type="text" value="mail"/>
Repository group:	<input type="text" value="--NONE--"/>
Alert repository group:	<input type="text" value="PINsafeUsers"/>

On the RADIUS > NAS screen, a single NAS will need to be defined, as follows:

NAS: Identifier:	<input type="text" value="NAS1"/>
Hostname/IP:	<input type="text" value="192.168.0.111"/>
Secret:	<input type="password" value="••••••"/>
EAP protocol:	<input type="text" value="None"/>
Group:	<input type="text" value="PINsafeUsers"/>

With this set up, to add a normal user to Swivel you just need to add them to the SwivelUsers group and synchronize. To create an Administrator, add a user to the SwivelAdministrators group and synchronize.

NOTE: When multiple Swivel servers are using the same repository it is very important that either the group definitions are the same for both Swivel servers, or if the servers are sharing the same database, that the repository is only defined on one of the Swivel servers. Users will still be able to authenticate on the other server.

## More Complex AD Group and Attribute example

In this example it is assumed that Swivel is being to authenticate a number of different users groups in a number of different methods.

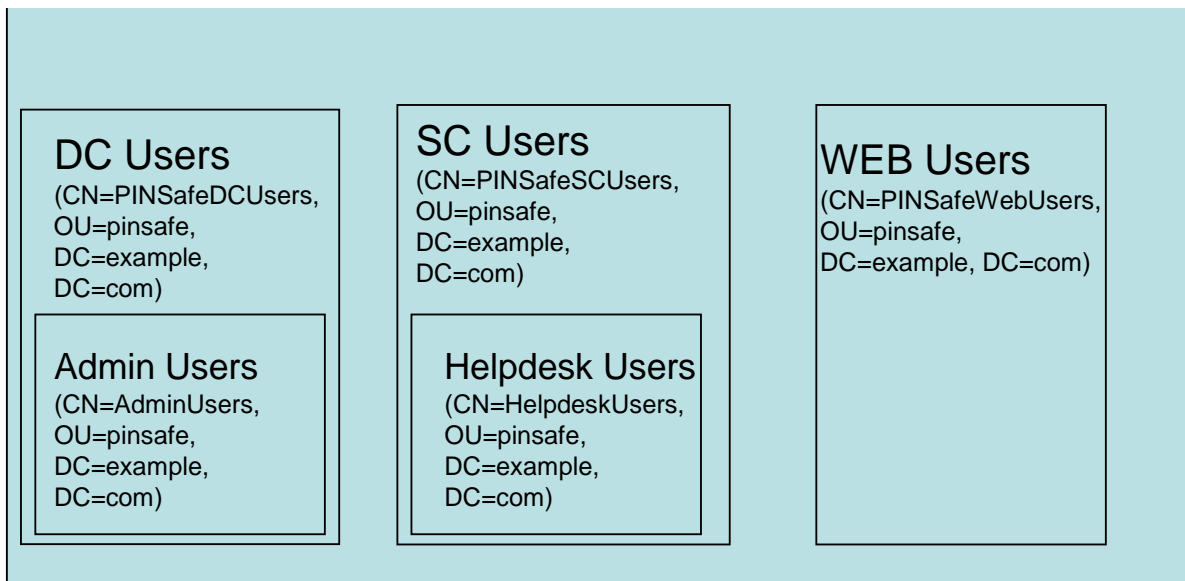
In this case Swivel is being used to authenticate access a website and a VPN. All users can use the VPN but only a subset of users can authenticate to the websites.

All users can use single channel authentication; some users can use dual-channel (SMS) authentication

Dual channel users will be sent alerts to their mobile phone, single channel users via email.

Swivel Administrators are dual channel users but Helpdesk users are single channel only.

This is a group structure that can support these requirements:



(If you are familiar with earlier versions of Swivel, note that in 3.3, there is no need to define a group that encloses all of the groups above).

The single channel group (SC Users) is the group which contains users that can only authenticate via single channel. In other words, SC Users and DC Users are distinct: no users are members of both of these groups, but all users must be members of one of these two groups.



The Admin users group can be a member of the dual-channel (DC) users group and the Helpdesk group can be a member of the SC Users group.

A separate group for those users that can authenticate to the website also needs to be created. This may contain members of both DC Users and SC Users.

With this group structure in place the Swivel server would use these groups in the following ways. On the repository -> groups page there would be the following settings

	Single	Dual	Swivlet	Admin	Helpdesk	PINless	
Name: <input type="text" value="DC Users"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<b>Definitions:</b>							<input type="button" value="Delete"/>
ith000140: <input type="text" value="DC Users"/>							
AD: <input "="" type="text" value="CN=PINSafeDCUsers, OU=pinsafe, DC=example, DC="/>							
Name: <input type="text" value="Admin Users"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<b>Definitions:</b>							<input type="button" value="Delete"/>
ith000140: <input type="text" value="Admin Users"/>							
AD: <input type="text" value="CN=AdminUsers, OU=pinsafe, DC=example, DC=cor"/>							
Name: <input type="text" value="SC Users"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<b>Definitions:</b>							<input type="button" value="Delete"/>
ith000140: <input type="text" value="SC Users"/>							
AD: <input "="" type="text" value="CN=PINSafeSCUsers, OU=pinsafe, DC=example, DC="/>							
Name: <input type="text" value="HelpDesk Users"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
<b>Definitions:</b>							<input type="button" value="Delete"/>
ith000140: <input type="text" value="HelpDesk Users"/>							
AD: <input "="" type="text" value="CN=HelpdeskUsers, OU=pinsafe, DC=example, DC="/>							
Name: <input type="text" value="WEB Users"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<b>Definitions:</b>							
ith000140: <input type="text" value="WEB Users"/>							
AD: <input "="" type="text" value="CN=PINSafeWebUsers, OU=pinsafe, DC=example, DC="/>							

**More complex repository groups example - group definitions**

Note that the WEB Users group does not have any rights defined on this page. It must be defined here, though, so that it can be used on other pages.

So that single channel users will have alerts, including on account creation, e-mailed to them, and dual channel users will have alerts sent to them via SMS, the Transport > General screen would include the following entries:

**for SMTP:**

Destination attribute	Mail
Repository Group	NONE
Alert Repository Group	SC Users

**for a GSM Modem (or SMS provider):**

Destination attribute	telephoneNumber
Repository Group	DC Users
Alert Repository Group	DC Users

To restrict website access to only those users that are in the WEB users group, the agent associated with the website would include the following entry.

Group	WEB Users
-------	-----------

Despite the variety of user-types, users can, for the most part, be added to Swivel just by making them a member of the appropriate AD group. The only exception is that users that need to be able to access via the website need to be added to the WEB Users group as well as either the Single Channel or Dual Channel group.

## Appendix D: Swivel Installation details.

<b>General</b>	
Date of Install	
Installation company name	
Engineer's name	
Customer Contact (inc email/phone)	
<b>Swivel Server Details</b>	
Appliance	Yes/No
Appliance serial number(s)	
High Availability	Yes/No
License Installed	Yes/No
Swivel version	
OS	
IP Address	
Default gateway	
DNS	
Language	English
RADIUS	Yes/No
Change PIN	Yes/No
User self reset.	Yes/No
Swivel Backup details	

### Support Arrangements

Hardware

OS

Software

### Customer Sign-off

The installation has been completed to my satisfaction in accordance with the details given above.

I understand the level of support I am entitled to as explained above.

Signature \_\_\_\_\_

Date: \_\_\_\_\_

Name \_\_\_\_\_

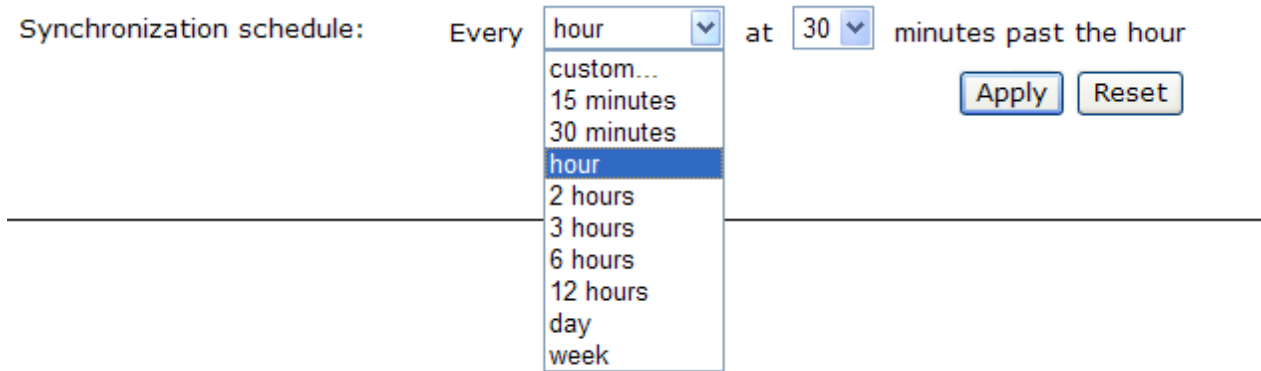
on Behalf of (company)

\_\_\_\_\_

## Appendix E: Setting Schedules and CRON Strings

With the exception of the session clean up setting, tasks can schedule using a *cron* syntax.

Most schedules can be set up by using the drop-down menu items on the synchronization setting screen.



If you select the custom option you can explicitly set the schedule using the *cron* syntax.

The use of the cron-like syntax gives a great deal of flexibility in scheduling these tasks. The settings require the following fields:

Field Name	Allowed Values	Allowed Special Characters
Seconds	0-59	, - * /
Minutes	0-59	, - * /
Hours	0-23	, - * /
Day-of-month	1-31	, - * ? / L W C
Month	1-12 or JAN-DEC	, - * /
Day-of-Week	1-7 SUN-SAT	- * ? / L C #
Year (Optional)	empty, 1970-2099	, - * /

These fields determine when the job is to be run. An asterisk (\*) in any field is a wildcard and means that the task will be run for all values of that field. A question-mark (?) means that the setting is not specified, this setting is applied to either the day-of-month and day-of-week as you cannot specify both of these parameters.

Therefore to schedule a job to run every hour, on the hour the settings would be

0 0 \* \* \* ?

Whereas to run a job 3 AM every Sunday would be

0 0 3 ? \* 1

## Document Versioning

3.1.4.1 October 2006

3.1.4.2 RADIUS and IP table advice added. 19th October 2006,

3.2 Manual for Swivel version 3.2. January 2007

3.3 Manual for Swivel version 3.3. July 2007

3.3 Update 1. November 2007

3.4 Manual for Swivel Version 3.4. April 2008

3.5 Manual for Swivel Version 3.5. October 2009

3.6 Manual for Swivel Version 3.6. June 2009

3.6.1 Minor amendments for Swivel Version 3.6.1. September 2009

3.7 Manual for Swivel Version 3.7. January 2010

3.8 Manual for Swivel Version 3.8. September 2010

3.9 Manual for Swivel Version 3.9. June 2012