Swivel Combined Client

Contents

Overview

This document describes a combined authentication and administration client for Swivel Server APIs which can be used in web servers using ASP.Net. It is a convenient wrapper around the XML-based APIs described here.

The client can also be used in non-web applications using the .Net Framework. See configuration below.

Prerequisites

The client DLL is available from here. This is version 1.3 of the client, which includes support for user attributes and for TLS protocol versions 1.1 and 1.2. This requires version 3.9.7 or later of the Swivel server. For TLS 1.1/1.2 support, version 3 of the Swivel appliance is required.

An example website demonstrating some of the features of the client is available from here. This download includes the DLL above, so you don't need to download both. Again, this test server uses the new client.

This version of the client requires Microsoft.Net Framework version 4.5 or later.

Using the Client

Configuring The Swivel Server

In order to use the client with a particular Swivel Server, you must define the computer running your web application as an Agent on that server. To do this:

- Log into the Swivel Server Administration Console
- Go to Server -> Agents
- In version 3.8 or later, expand the entry at the bottom labelled "New Entry". In earlier versions, this will already be visible.
- Enter a name for the new Agent. Note that any users created using the API will be added to a repository with this name, so the name should not be the same as an existing repository if you intend to manage users through this Agent.
- Enter the IP address of the web server under Host/IP.
- If you will be managing users through this Agent, set "Can act as Repository" to Yes.
- The remaining entries can be left as default for now. Click Apply.

Depending on how you intend to use the client, you may like to create a new Repository Group for users created by the Agent, or you may prefer to use an existing Group. You can create a new Group under Repository -> Groups. All you need to add is a name in the blank entry at the bottom. There is no need to add definitions for other repositories, or to specify rights for this Group, as you will be specifying user rights explicitly when the user is created. You may also like to go back to the Agent definition and set the Group for this Agent to the Group you have just created. If you do this, only users created by the Agent will be able to authenticate through the Agent.

If you have created a new Group, you should also set up Transports for this Group. You should at least set up an Alert Transport, and if you are using dual channel authentication, a Strings Transport as well. Note that if you want to use an existing Transport, you will need to copy the class name to a new Transport and give it a new name. Select the Group you have just created as the Strings or Alert Group as appropriate.

Configuring Your ASP.Net Application

In order to use the client, you need to add certain entries to the web.config file in your ASP.Net application. The following is an example:

```
 <appSettings>

   <add key="PINsafeServer" value="myserver"/>
   <add key="PINsafePort" value="8080"/>
   <add key="PINsafeContext" value="sentry"/>
   <add key="PINsafeSecret" value="secret"/>
   <add key="PINsafeSecure" value="True"/>
   <add key="PINsafeAcceptSelfSigned" value="False"/>
   <add key="AllowNonPINsafeUsers" value="False"/>
   <add key="IgnoreDomainPrefix" value="True"/>
   <add key="IgnoreDomainSuffix" value="False"/>
   <add key="PINsafeAgentVersion" value="3.97" />
   <add key="PINsafeTlsProtocols" value="Tls12" />

 </appSettings>
```

The comments at the start and end are not necessary - they are for clarity - but there is a method in the client to remove these settings which requires these comments. The <appSettings> element should be contained within the main <configuration> element, and may already exist, depending on your application.

If you are using the client in an executable (i.e. non-web) application, you can use exactly the same settings, but these must be in a file named *Application*.exe.config, where *Application* is the name of the executable application. This must be in the same directory as the program executable.

The meanings of the key names are shown below:

- **PINsafeServer** - The host name or IP address of the Swivel server
- **PINsafePort** - The port used by the Swivel server, normally 8080
- **PINsafeContext** - The context (application URL) of the Swivel server, normally "pinsafe"
- **PINsafeSecret** - The shared secret as configured in the Agent entry previously
- **PINsafeSecure** - "True" if HTTPS is to be used for communication with, "False" otherwise
- **PINsafeAcceptSelfSigned** - "True" if SSL certificate errors should be ignored

- **AllowNonPINsafeUsers** - "True" if users unknown to PINsafe should be authenticated automatically, "False" if they should be rejected
- **IgnoreDomainPrefix** - "True" if the domain prefix in usernames such as domain\user should be stripped off before checking with Swivel server
- **IgnoreDomainSuffix** - "True" if the domain suffix in usernames such as user@domain should be stripped off before checking with Swivel server
- **PINsafeAgentVersion** - Sets the *Version* attribute sent with the API request. The default is "3.4". To support user attributes, this should be set to "3.97".
- **PINsafeTlsProtocols** - Specifies which TLS protocols are supported. You can specify Ssl3, Tls1, Tls11 or Tls12. Separate supported protocols with commas. The default is "Tls1,Tls11,Tls12".

NOTE: on an appliance, you should not use the proxy application (and port 8443), as functionality in the proxy is deliberately limited. You must use the pinsafe application to get the full functionality out of the client.

If it is not practical to provide the settings in a configuration file, you can create an instance of the SwivelSettings class, and initialise it as follows:

```
SwivelSettings swivelSettings = SwivelSettings.EmptySettings;
swivelSettings.Server = "myserver";
swivelSettings.Port = 8080;
swivelSettings.Context = "sentry";
swivelSettings.Ssl = true;
swivelSettings.Secret = "secret";
swivelSettings.AcceptSelfSigned = false;
swivelSettings.TlsProtocols = SecurityProtocolType.Tls12;
```

You can then use this instance when creating a request - see the class documentation for details.

Implementing the Swivel Client in ASP.Net Applications

In order to use the client DLL, you need to copy it to the Bin folder of your ASP.Net application.

Technical documentation for the client is in progress. Please see the example application.

The namespace for all classes in the client is **swivelsecure.client**.

Documentation of the authentication class can be found here

The user management classes fall into two categories: administrator and helpdesk. The administrator methods are more extensive, allowing you to create, update and delete users. However, they are limited to users belonging to a single repository, named after the Agent corresponding to the computer the application is running on. The helpdesk methods are more limited, typically read-only methods, but with some update functionality. However, these methods can apply to users in other repositories. See the two documents for details.