

UserExchange Web Service Custom Application Integration Guide

Introduction

Use this guide to enable Multi-Factor Authentication and Single Sign-on (SSO) access via encryption and web service to customer-controlled web applications.

Workflow:

1. End-users will be redirected (via browser) to SecureAuth IdP for authentication upon attempting access into the application without an **internal session token** or an **encrypted string** from SecureAuth
2. SecureAuth IdP will then authenticate the end-user, and send the **encrypted authenticated User ID** in a **cookie, query string, or the header** (options represented as **token** in Custom Application Configuration Steps) to the application following a successful authentication
3. The application will detect the encrypted string, **extract** it, and then contact the **SecureAuth IdP web service** to retrieve the User ID in **plaintext**
4. The application then uses the User ID to create an **internal session token** and to **authorize** the user into the application

Prerequisites

1. Have a custom web application and modification access
2. Enable the UserExchange Web Service



1. On the SecureAuth IdP appliance, open `D:\SecureAuthWS\web.config`
2. Locate the key, **EnablePOC**
3. Change the value of the key to 1

```
<add key="EnablePOC" value="1" />
```

3. Create a **New Realm** for the custom application integration in the SecureAuth IdP Web Admin
4. Configure the following tabs in the Web Admin before configuring the **Post Authentication** tab:
 - **Overview** – the description of the realm and SMTP connections must be defined
 - **Data** – an enterprise directory must be integrated with SecureAuth IdP
 - **Workflow** – the way in which users will access this application must be defined
 - **Multi-Factor Methods** – the Multi-Factor Authentication methods that will be used to access this page (if any) must be defined

Custom Application Configuration Steps

1. Change the application's **Login Page** to look for some type of **internal session token**

This will alert the application to allow access to end-users if it is present (bypassing additional authentication)

2. If there is no internal session token, have the **Login Page** look to see if there is a **Token Name, eu** (the cookie, query string, or header from SecureAuth IdP)

3. If there is no **eu** token, enable the redirection to the **SecureAuth IdP URL**, along with the **ReturnURL** as a **query string**

Example redirect URL: **https://secureauth.company.com/secureauth2?ReturnURL=https://app.company.com**

The **SecureAuth IdP URL** will be the Fully Qualified Domain Name (FQDN) of the SecureAuth IdP appliance, followed by the SecureAuth IdP custom application-integrated realm (**https://secureauth.company.com/secureauth2**)

The **ReturnURL** will typically be the FQDN of the custom application

4. If there is an **eu** token, extract the value inside of the token

5. Make a web service call to retrieve the **Authenticated User ID** value by passing in the value of the **eu** token as **input parameter** to the function

A copy of the web service is located at the FQDN of the SecureAuth IdP appliance, followed by **/SecureAuthWS/UserService.asmx**, e.g. **https://secureauth.company.com/SecureAuthWS/UserService.asmx**

A copy of the WSDL for development is located at the FQDN of the SecureAuth IdP appliance, followed by **/SecureAuthWS/UserService.asmx?WSDL**, e.g. **https://secureauth.company.com/SecureAuthWS/UserService.asmx?WSDL**

The **GetUserID** takes **one input parameter**, the **EncryptedUserString** (the **eu** token value)

This will return the **Decrypted Authenticated User ID** as a **String** regardless of when the token was generated

The **GetUserIDWithTimeCheck** takes **two input parameters**, **EncryptedUserString** and **ValiditySeconds**

This will only return the **Decrypted Authenticated User ID** as a **String** if the token was generated within the validity time frame (e.g. 5 - 10 seconds)

6. Generate an **internal session token**, and take the **Decrypted Authenticated User ID String** and log the user into the application

7. **(OPTIONAL)** Clear the value inside the **eu** token so that it cannot be reused

SecureAuth IdP Configuration Steps

▼ Profile Fields

Property	Source	Field	Data Format	Writable
Groups	Default Provider	memberOf		<input type="checkbox"/>
First Name	Default Provider	givenName		<input type="checkbox"/>
Last Name	Default Provider	sn		<input type="checkbox"/>
Phone 1	Default Provider	telephoneNumber		<input checked="" type="checkbox"/>
Phone 2	Default Provider	mobile		<input checked="" type="checkbox"/>
Phone 3	Default Provider	homePhone		<input checked="" type="checkbox"/>
Phone 4	Default Provider	Pager		<input checked="" type="checkbox"/>
Email 1	Default Provider	mail		<input checked="" type="checkbox"/>
Email 2	Default Provider	Application ID		<input type="checkbox"/>

1. In the **Profile Fields** section, map the directory field that contains the user's Application ID to the SecureAuth IdP **Property**

For example, add the Application ID to the **Email 2 Property** if it is not already contained somewhere else



Click **Save** once the configurations have been completed and before leaving the **Data** page to avoid losing changes

Post Authentication

▼ Post Authentication

Authenticated User Redirect:

Redirect To:

Upload a Page: No file chosen

[Download Customized Pages](#)

2. Select **User Handler Web Service** from the **Authenticated User Redirect** dropdown in the **Post Authentication** tab in the Web Admin

3. An unalterable URL will be auto-populated in the **Redirect To** field, which will append to the domain name and realm number in the address bar (Authorized/EncryptUser.aspx)

User ID Mapping

▼ User ID Mapping

User ID Mapping: Transformation Engine

Name ID Format:

Encode to Base64:

4. Select the SecureAuth IdP **Property** that corresponds to the directory field that contains the Application ID (**Email 2**)

URL Redirect

▼ URL Redirect

URL:

PostAuth Webservice URL:

5. Set the **URL** to the application URL to where the end-user will be redirected upon successful authentication, e.g. **https://app.company.com**

User Encryption Key

▼ User Encryption Key

Key Value:

Transport Name:

Transport Method:

- Query String
- ✓ Cookie
- Header

6. Leave the **Key Value** field blank, or if there is a value present, then remove it

7. Set the **Transport Name** to a different name than the defaulted **eu** if preferred

If the **Transport Name** is changed here, then it will need to be changed in the application as well

Modify the Custom Application Configuration Steps to replace the **eu** value with the new **Transport Name** value

8. Select how the **Authenticated User ID** will be transported from the **Transport Method** dropdown

 Click **Save** once the configurations have been completed and before leaving the **Post Authentication** page to avoid losing changes

Forms Auth / SSO Token

▼ Forms Auth/SSO Token

Key Generation: [View and Configure FormsAuth keys/SSO token](#)

9. Click **View and Configure FormsAuth keys / SSO token** to configure the token/cookie settings and to configure this realm for SSO

 These are *optional* configurations

▼ Forms Authentication

Name: Login Uri: Domain: Require SSL: Cookieless: Sliding Expiration: Timeout:

Minute(s)

1. If SSL is required to view the token, select **True** from the **Require SSL** dropdown
2. Choose whether SecureAuth IdP will deliver the token in a cookie to the user's browser or device:
 - **UseCookies** enables SecureAuth IdP to always deliver a cookie
 - **UseUri** disables SecureAuth IdP to deliver a cookie, and instead deliver the token in a query string
 - **AutoDetect** enables SecureAuth IdP to deliver a cookie if the user's settings allow it
 - **UseDeviceProfile** enables SecureAuth IdP to deliver a cookie if the browser's settings allow it, no matter the user's settings
3. Set the **Sliding Expiration** to **True** if the cookie remains valid as long as the user is interacting with the page
4. Set the **Timeout** length to determine for how many minutes a cookie is valid



No configuration is required for the **Name**, **Login URL**, or **Domain** fields

Machine Key

Machine Key

Validation:

Decryption:

Validation Key:

Decryption Key:

5. No changes are required in the **Validation** field, unless the default value does not match the company's requirement

If a different value is required, select it from the dropdown

6. No changes are required in the **Decryption** field, unless the default value does not match the company's requirement

If a different value is required, select it from the dropdown



No configuration is required for the **Validation Key** or **Decryption Key** fields

Authentication Cookies

▼ Authentication Cookies

Pre-Auth Cookie:

Post-Auth Cookie:

Persistent:

Clean Up Pre-Auth Cookie:

7. Enable the cookie to be **Persistent** by selecting **True - Expires after Timeout** from the dropdown

Selecting **False - Session Cookie** enables the cookie to be valid as long as the session is open, and will expire once the browser is closed or the session expires

 No configuration is required for the **Pre-Auth Cookie**, **Post-Auth Cookie**, or the **Clean Up Pre-Auth Cookie** fields

Click **Save** once the configurations have been completed and before leaving the **Forms Auth / SSO Token** page to avoid losing changes

 To configure this realm for SSO, refer to [SecureAuth IdP Single Sign-on Configuration](#)

 To configure this realm for *Windows Desktop SSO*, refer to [Windows desktop SSO configuration](#)