

Relius Administration Web Single Sign-On (SSO)

Technical Manual

(SAML 2.0 and Legacy Technologies)

For Public Consumption
Last Updated: September 11, 2018
Relius Technology Support

Empowering
the Financial World



Contents

Overall Purpose	4
Single Sign-On SAML 2.0 System Overview	4
Assumptions	4
User Profiles.....	4
Licensing	5
Password Expiration	5
Timeouts.....	5
Landing Pages	5
Custom Error Messages	5
Affected Sites.....	5
Generic Inbound SAML SSO Setup.....	6
Current Modify Advice/Provider Screen	6
Client Public Certificate Setup in Relius Administration	7
Private Certificate Setup in Relius Administration	8
SAML SSO Posting details	9
Assertion page.....	9
Page to which SAML request is posted for participants.....	9
Participant web	9
Sponsor /Advisor web.....	9
New Method to fetch client certificates	9
Query string Data:	9
SAML Assertion fields.....	9
Assertion parameters used for participant web:	9

Token parameters used for Sponsor Web / Advisor Web:..... 10

Sample SAML2.0 XML Assertion..... 14

Participant:..... 14

Participant SAML SSO Request Mapping: 21

Sponsor SAML SSO Request Mapping: 26

For Internal testing (Note the Client Private certificate and Relius public key must reside in the \reliusweb\SSO of the web server to use this test page). 28

You can test by using testpage as mention below: 28

Server/Website Name/testsaml2.aspx 28

For Participant Inbound: 28

For Sponsor Inbound: 29

Certificate Generation Utility 30

Single Sign-On Legacy System Overview 45

Legacy Single Sign On Steps 46

Error Handling..... 46

Token Fails Validation..... 47

Session Expiration 47

Communication Protocols 47

Token Creation..... 47

Token Generation 48

Token Response 49

Client to SSO Web Service 49

Form Post to SSO page..... 49

Sample HTTP POST Request Form..... 50

Settings for Legacy SSO 50

Web settings. 50

Optional Field Variables	51
TokenGenerator WSDL	51
©2015 FIS and/or its subsidiaries. All Rights Reserved	51

Overall Purpose

This document provides setup information and samples of our Relius Administration Single Sign On Product for the Relius Web Application. It covers the SAML 2.0 technology and SSO implementation using self-sign certificates. It also details the setup process using the legacy technology of generating a token to authenticate the SSO process.

Single Sign-On SAML 2.0 System Overview

Generic SAML 2.0 SSO for participant, sponsor, and advisor web. The purpose of this document is to describe configuration requirements of the Single Sign-On (SSO) component for the Relius Web application. The SSO component provides the mechanism for allowing the inbound single sign-on using SAML 2.0 technology.

The only requirements are two sets of 64 encoded X509 (256 bit) certificates. You can create self-sign certificates or procure these from a certificate authority. Please make sure to ask for the 256 bit certificate if you're going to use the certificate authority. For self-sign certificates, there are instructions at the end of the SAML SSO Generic Inbound section.

It will also cover the legacy SSO setup without using the SAML 2.0 technology.

This document is to be distributed only to clients of Relius Administration who have purchased the SSO feature of the Relius Administration Web module. Unauthorized use or distribution is prohibited. The procedures outlined herein are designed for an advanced user, specifically for Web developers and engineers. While there is no harm in distributing to clients in the field, caution should be observed in ensuring that the user is proficient in IIS, security protocols, networking systems, and the responsibilities of administrative privileges.

Assumptions

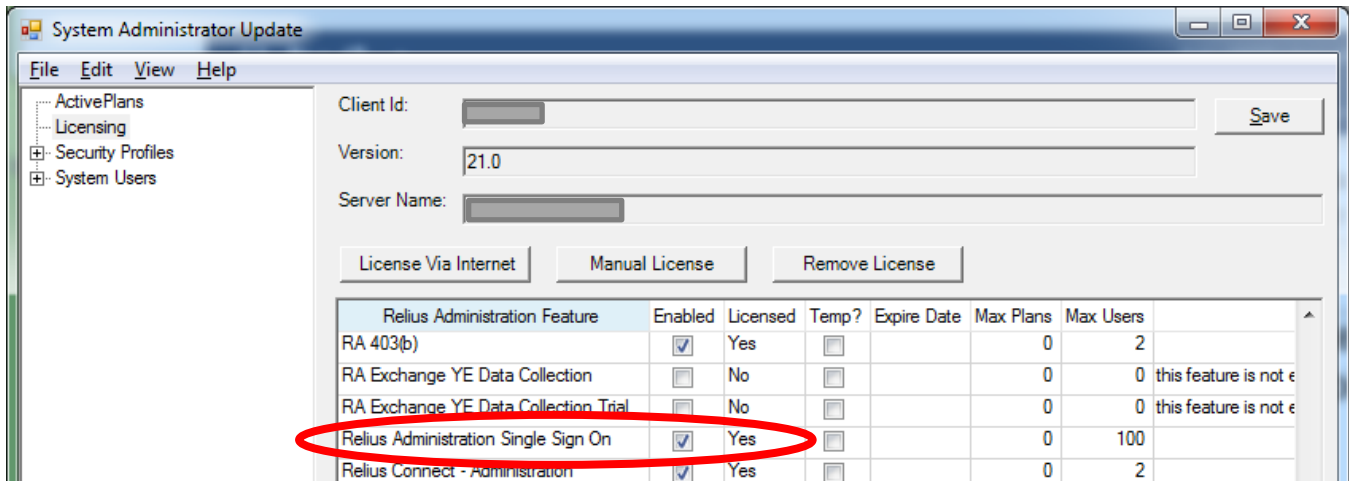
To get SSO to work properly, there are several assumptions that must be met.

User Profiles

For every user that logs into ReliusWeb via SSO, that user (including participants, plan sponsors, and advisors) will need to have an active user profile in the system. In other words, the user will need to be successfully integrated to the ReliusWeb site before being able to sign in via SSO.

Licensing

Clients are assumed to have been licensed for Relius Administration Single Sign-On. If you do not see this in your licensing module, please contact your Sales Representative.



Password Expiration

SSO is not affected by password security settings, including complexity and expiration. The SSO function authenticates users only with the token that is generated.

Timeouts

As the ReliusWeb page times out due to inactivity, the user signed in via SSO will be redirected back to the client's page from which the user entered the ReliusWeb site. This will be the trueLoginPage if it's provided. If there's no trueLoginPage provided the system will land the user on the request page (i.e. ParticipantSingleSignon.aspx).

Landing Pages

Clients can set what pages users will arrive at when they are done with the SSO portion. See the <TRUELOGINPAGE> and <ERRORLOGINPAGE> settings later in the document.

Custom Error Messages

Due to security restrictions, Relius SSO does not pass much information on errors. However, clients are free to create an "Error Landing" page to receive the error feedback that is generated when the token is returned as invalid.

For example, the error message when an invalid token is passed back with an invalid SSN has the following description: "DESCRIPTION=Login+failed.+Invalid+userid%2fpassword+combination." This data can then be fed into the custom error page on the client's primary site.

Affected Sites

SSO is valid for logging into the Participant, Sponsor, or Advisor Web sites.

©2018 FIS and/or its subsidiaries. All Rights Reserved.

Generic Inbound SAML SSO Setup

Current Modify Advice/Provider Screen

(RA->Utilities->VRU/Web Administrator->Maintenance->Utilities->Advise Provider->Setup)

In General information groupbox, added a read only text field labeled "APID:" in the Modify provider form.

Note:The below example use the **Generic** Provider. If you have our RA Connect product and used "**Generic**" Provider to set up the adviceprovider config files for connect do not use the "**Generic**" provider here for SSO. Select a different provider (format / TPAID).

Note: To pass APID field from Query string this field is displayed on Modify Advice/Provider screen so that client can pass it.

Client Public Certificate Setup in Relius Administration

In the same screen, attach the client public certificate used in the SAML assertion. Browse for the public certificate (.cer file) on your local drive.

Modify Advice/Provider

General information

Format:

TPA ID:

AP ID:

FTP information

Server:

User Id:

Password:

Certificate

Certificate:

Password:

Certificate information:

Property	Value
Issuer Name	CN=localhost
Subject Name	CN=localhost

Single sign-on

Client ID:

Password:

Private Certificate Setup in Relius Administration

The TPAs private certificate should be uploaded in the TPA Information interface (Utilities>System Administration>TPA Information). Browse out to the .pfx file and enter the associated password. Click on load certificate to view the certificate's properties.

TPA Information

TPA/Company

Select Company: SunGard Relius

Address

Street 1: 1660 Prudential Drive

Street 2:

City: Jacksonville

State: FL Foreign state/prov:

Postal Code: 32207 Country:

Phone

Primary pho

Primary fax:

Broadridge

Master clie

ACH information

Financial Institution

Originating DFI

Immediate Destination

Immediate Origin

Certificate

Certificate: ReliusPrivate.pfx

Password: *****

Certificate information:

Property	Value
Issuer Name	CN=www.idp.com
Subject Name	CN=www.idp.com
Expires on	12/31/2049 9:00:00 AM

SAML SSO Posting details

Assertion page

Page to which SAML request is posted for participants.

Participant web

AssertionConsumerService.aspx?APID=G is the page to which SAML SSO login requests for participant web should be posted.

Sponsor /Advisor web

psAssertionConsumerService.aspx?APID=G is the page to which SAML SSO login requests for sponsor or advisor web should be posted.

New Method to fetch client certificates

To support generic client SAML request to Relius Web, we added method in AdviceProvider Class (in Relius.Admin.Bus.Links DLL), which fetches the Client public certificate based on respective APID (1). To add the Relius private certificate, the TPAs private certificate should be uploaded in the TPA Information interface (See Section 1 above).

Query string Data:

The following parameters can be passed in the URL

- APID: **Required.** decryptassertions: true/false
- (Optional) validateinboundtokenexpiration : true/false (Optional)

--for debugging SAML token. For client, not mandated. When SAML token is issued, there is an expiration.... When this variable is set to FALSE

SAML Assertion fields

Assertion parameters used for participant web:

Field	Value	Required/Optional	Purpose
SSN	PERSON.SSNum	Required	Participant lookup
WEBCOOKIE	Person.WebCookie	Optional	Optional, alternate method to lookup SSN. Links to Person.WebCookie
LOGINTYPE	P	Optional	Indicates request is for participant web

FIRSTPAGE	Page name (i.e. transfer.aspx)	Optional	Used to land the user on a specific page. If user is in more than one plan, they will also need to specify PLANID to utilize this value
TRUELOGINPAGE	URL to return employee on logout	Optional	Used to land the user somewhere specific on logout
PLANID	Relius.Planid	Optional	Can be used to bypass plan selection page for participants in multiple plans. If not utilized, if the participant is in more than one plan, land on the plan selection page to determine the plan
ERRORLOGINPAGE	URL used for error handling	Optional	Specify a landing page in the event of an error
LANGUAGE	ENG or SPA	Optional	Assumption is English. If Spanish is specified, open the web in the Spanish language mode

Token parameters used for Sponsor Web / Advisor Web:

Field	Value	Required/Optional	Purpose
CONTACTID	Contact.CONTACTID	Required	Used to look up the contact that is logging in. If UpdateContact is set to True, then update the existing sponsor user, else create new. For creating a new sponsor user not in the system, the recommendation is varchar (38) guid value
LOGINTYPE	S or A	Optional	S = Sponsor Web A = Advisor Web

FIRSTPAGE	Page name (i.e. DVC.aspx	Optional	Used to land the user on a plan specific page. If this is a plan or participant specific page, then will require PLANID/SSN values passed in to support
TRUELOGINPAGE	URL to return EE on logout	Optional	Used to land the user somewhere specific on logout
PLANID	Relius.PlanId	Optional	Used to login to a specific plan
ERRORLOGINPAGE	URL used for error handling	Optional	Used to specify a landing page in the event of an error
SSN	Participant SSNum	Optional	Used in conjunction with PLANID to log the sponsor into a participant page in sponsor mode
UPDATECONTACT	True or False, assumption is False	Optional	Required only when sending in UPDATECONTACT values
DVCCODES			
REQUESTID	Dvcimportmstr.RequestId for previous payroll process	Optional	Use with DVC to open an existing payroll
DVCSUBMITFINAL	URL	Optional	URL to which the user is redirected when sponsors have access to submit for final processing
Field	Value	Required/Optional	Purpose
DVCNOSUBMITFINAL	URL	Optional	URL to which the user is redirected when they do not have access to submit for final processing
PAYENDDATE	Payroll end date as DDMON-YYYY (i.e. 08FEB2017)	Optional	Used to open an existing DVC payroll

PAYFREQCD	Payroll Frequency associated with the DVC records (i.e. A, W, B, S, etc.)	Optional	Used to open an existing DVC payroll
UPDATECONTACT set True (Note: Currently only available for Sponsor)			
ErlId	ErlId Value	Optional	The company ID associated with the contact. <i>Note that this field is required when creating a new sponsor not in the system</i>
FirstName	First Name	Optional	This is the first name of the contact as displayed on the web
LastName	Last Name	Optional	This is the last name of the contact as displayed on the web
EmailAddress	Email Address	Optional	This is the email address associated with the sponsor
UserId	User Id	Optional	May be used to set or update a user's ID used to login from the standard Relius login page
Password	Password	Optional	May be used to update a user's password used to login from the standard Relius login page

Field	Value	Required/Optional	Purpose
AllPlanAccessCd	AllPlanAccessCd	Optional	This is a new optional BOOLEAN variable that indicates whether the sponsor has access to all available plans in the database with sponsor enabled. If this code comes over as True, we will insert a Plan contact association for each plan. This code should be False if records are submitted in the AccessByPlan node
AllPlanAccessSecProfileId	AllPlanAccessSecProfileId	Optional	If AllPlanAccessCd is sent in as TRUE, this value must be specified. This will be used to update the contact's security profile for all plans in the database to the specified value
AccessByPlan	AccessByPlan	Optional	Create a sub-node that can be used to populate records that will give access to the contact for specific plans. Each record should have PlanId and SecProfileId. Format: PlanId SECURITYPROFILEID; Example: 123 456;789:000;...

Sample SAML2.0 XML Assertion

Participant:

```

<saml:Response ID="_d8e2c81f-3b4c-40dc-9e3e-6baa47d22e48" Version="2.0" IssueInstant="2018-07-30T12:49:03.392Z"
Destination="http://localhost/reliusadminweb_20180_sasi1/AssertionConsumerService.aspx?clientid=&apld=G"
xmlns:saml="urn:oasis:names:tc:SAML:2.0:protocol">

  <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">www.relius.com</saml:Issuer>

  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">

    <SignedInfo>

      <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />

      <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />

      <Reference URI="#_d8e2c81f-3b4c-40dc-9e3e-6baa47d22e48">

        <Transforms>

          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />

          <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">

            <InclusiveNamespaces PrefixList="#default saml ds xs xsi" xmlns="http://www.w3.org/2001/10/xml-exc-c14n#" />

          </Transform>

        </Transforms>

        <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />

        <DigestValue>ukSYPMBvb4IR4Q3sB5Da6URoqY4=</DigestValue>

      </Reference>

    </SignedInfo>

    <SignatureValue>Ld0e9UTvXw1W2nWCRtW2zNuO/CV8Q5NoMBR4rOaPaNz1grH9dNV7OV2u6IPGpnswn1xVn6vh
XQDNDRpLbIZmoiXenG1dxT2rAhkF75V1joUje3n3SmoBUU036FIASFGjmSFRDuRKciZTKda23PiNvOtr0JmN2f55N5
xgDJUxc5c=</SignatureValue>

    <KeyInfo>

      <X509Data>

        <X509Certificate>MIIB7jCCAVugAwIBAgIQokPwiChvlohBoVBXqgqwHzAJBgUrDgMCHQUAMBQxEjAQBgNVBAMT
CWxvY2FsaG9zdDAeFw0wMDAxMDEwMzAwMDEwMzAwMDBaFw0zNjAxMDEwMzAwMDEwMzAwMDBaMBQxEjAQBgNVBAMTCWxvY2Fs
aG9zdDCBnzANBjGkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAz3cBadlr+pY/ueXDZuQZktqrd8h9BBWix5sQ89yNr7zU9I
+84Osr3XUw/Fi2HIQa9jkEBPXvHShDpLGjLwfPas3FWlrHsPjmj1iY3BzmDyY0oTOT5i4PypVbnrFKK/tqCStxaZa+tsc

```

```
FNwGjNRhV01axlcFCd69kzTM4rI741sCAwEAAaNJMEcwRQYDVR0BBDD4wPIAQWM9ncWWDz4IYgOJA6ydyccqEW
MBQxEjAQBgNVBAMTCWxvY2FsaG9zdIIQokPwiChviahBoVBXqgwwqHzAJBgUrDgMCHQUAA4GBALK3PUvLORkG
eDmjPghj4YUrheUhUqQe8dvNNBXmt4NzSGGbhPcqvaxG3WOadsXRnzINyI3fJU5vyQyz8OCSSep3uClL4Ui8dbcq
22TQfUlk8LKPHN9XWWUXOMlnTe8NAD4X1QqX9BILKqFLZl6J3rjRishFMnF5Kp6YyBRqpw3</X509Certificate>
```

```
</X509Data>

</KeyInfo>

</Signature>

<samlp:Status>

  <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success" />

</samlp:Status>

<saml:Assertion Version="2.0" ID="_ea4cc5f7-4609-4e5e-9b9e-ec97dd8f6032" IssueInstant="2018-07-
30T12:49:03.392Z" xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">

  <saml:Issuer>www.relius.com</saml:Issuer>

  <saml:Subject>

    <saml:NameID></saml:NameID>

    <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">

      <saml:SubjectConfirmationData NotOnOrAfter="2018-07-30T12:54:03.392Z"
Recipient="http://localhost/relusadminweb_20180_sasi1/AssertionConsumerService.aspx?clientid=&apld=G" />

    </saml:SubjectConfirmation>

  </saml:Subject>

  <saml:Conditions NotBefore="2018-07-30T12:49:03.392Z" NotOnOrAfter="2018-07-30T12:54:03.392Z" />

  <saml:AuthnStatement AuthnInstant="2018-07-30T12:49:03.392Z" SessionNotOnOrAfter="2018-07-
30T12:54:03.392Z">

    <saml:AuthnContext>

<saml:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:Password</saml:AuthnContextClassRef>

    </saml:AuthnContext>

  </saml:AuthnStatement>

  <saml:AttributeStatement>

    <saml:Attribute Name="APID"> <!-- Mandatory -->

      <saml:AttributeValue>G</saml:AttributeValue>

    </saml:Attribute>

    <saml:Attribute Name="SSN"> <!-- Mandatory -->
```

©2018 FIS and/or its subsidiaries. All Rights Reserved.

```
<saml:AttributeValue>000881212</saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="LoginType">
  <saml:AttributeValue>P</saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="FirstPage">
  <saml:AttributeValue>editperson.aspx</saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="TrueLoginPage">
  <saml:AttributeValue>default.aspx</saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="ErrorLoginPage">
  <saml:AttributeValue>error.aspx</saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="PlanID">
  <saml:AttributeValue>87234</saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="LANGUAGE">
  <saml:AttributeValue>ENG</saml:AttributeValue>
</saml:Attribute>
</saml:AttributeStatement>
</saml:Assertion>
</samlp:Response>
```


Sponsor:

```

<samlp:Response ID="_aa751a7c-2761-46e9-879f-265c60e5b808" Version="2.0" IssueInstant="2018-07-30T12:54:45.232Z"
Destination="http://localhost/reliuadminweb_20180_sasi1/psAssertionConsumerService.aspx?clientid=&apld=G"
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol">

  <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">www.relius.com</saml:Issuer>

  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">

    <SignedInfo>

      <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />

      <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />

      <Reference URI="#_aa751a7c-2761-46e9-879f-265c60e5b808">

        <Transforms>

          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />

          <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">

            <InclusiveNamespaces PrefixList="#default samlp saml ds xs xsi" xmlns="http://www.w3.org/2001/10/xml-exc-c14n#" />

            </Transform>

          </Transforms>

          <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />

          <DigestValue>LwP4I0A8byhC3TxxVkedRQJTJD0=</DigestValue>

        </Reference>

      </SignedInfo>

      <SignatureValue>JKYZCtyfrCLF+RBAdicw80nY3zXZAsX2kFu0r7ceyJY2hSnRC3Hol9sYmL1JOP8AG1V3/q5OEukg
AJuYyaq/4Gzg1fE/FXWA0slbHIEY+3yyJ66qtlvUk0DPxkri0Wg4T5UEr8bDAooEvh/6rgjd3cGbJDUMcuKbygJQIQi5dRs
=</SignatureValue>

      <KeyInfo>

        <X509Data>

          <X509Certificate>MIIB7jCCAVugAwIBAgIQokPwiChviahBoVBXqgqwHzAJBgUrDgMCHQUAMBQxEjAQBgNVBAMT
CWxvY2FsaG9zdDAeFw0wMDAxMDEwMzAwMDEwMzAwMDBaFw0zNjAxMDEwMzAwMDEwMzAwMDBaMBQxEjAQBgNVBAMT
CWxvY2FsaG9zdDCBnzANBjGkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAAt3cBadlr+pY/ueXDZuQZktqrd8h9BBWix5sQ89yNr7zU9I
+84Osr3XUtW/Fi2HIQa9jkEBPXvHShDpLGjLwFPas3FWlrHsPjmj1iY3BzmDyY0oTOT5i4PypVbnrFKK/tqCStxaZa+tsc
FNwGjNRhV01axlcFCd69kzTM4rI741sCAwEAQAaAJMEcwRQYDVR0BBBD4wPIAQWm9ncWWDz4IYgOJA6ydyqcqEW
MBQxEjAQBgNVBAMTcwY2FsaG9zdIIQokPwiChviahBoVBXqgqwHzAJBgUrDgMCHQUAA4GBALK3PUvLORkG

```

```

eDmjPghj4YUrheUhUqQe8dvNNBXmt4NzSGGbhPcqvaXG3WOadsXRnzINYI3fJU5vyQyz8OCSSep3uCkL4Ui8dbcq
22TQfUIk8LKPHN9XWWUXOMInTe8NAD4X1QqX9BILKqFLZl6J3rjRishFMnF5Kp6YyBRqpw3</X509Certificate>

  </X509Data>

</KeyInfo>

</Signature>

<samlp:Status>

  <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success" />

</samlp:Status>

<saml:Assertion Version="2.0" ID="_04aba2a6-ae1d-418e-85aa-df214d7e00b5" IssueInstant="2018-07-
30T12:54:45.232Z" xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">

  <saml:Issuer>www.relius.com</saml:Issuer>

  <saml:Subject>

    <saml:NameID></saml:NameID>

    <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">

      <saml:SubjectConfirmationData NotOnOrAfter="2018-07-30T12:59:45.232Z"
Recipient="http://localhost/reliusadminweb_20180_sasi1/psAssertionConsumerService.aspx?clientid=&apld=G"
/>

    </saml:SubjectConfirmation>

  </saml:Subject>

  <saml:Conditions NotBefore="2018-07-30T12:54:45.232Z" NotOnOrAfter="2018-07-30T12:59:45.232Z" />

  <saml:AuthnStatement AuthnInstant="2018-07-30T12:54:45.232Z" SessionNotOnOrAfter="2018-07-
30T12:59:45.232Z">

    <saml:AuthnContext>

<saml:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:Password</saml:AuthnContextClassRef>

    </saml:AuthnContext>

  </saml:AuthnStatement>

  <saml:AttributeStatement>

    <saml:Attribute Name="ContactID"> <!-- Mandatory -->

      <saml:AttributeValue>0924EBEB701C4A8AA30D0E1CC0C34C90</saml:AttributeValue>

    </saml:Attribute>

    <saml:Attribute Name="APID"> <!-- Mandatory -->

```

```
<saml:AttributeValue>G</saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="SSN">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="LoginType"> <!-- Mandatory -->
  <saml:AttributeValue>S</saml:AttributeValue><!-- S: Sponsor, A:Advisor-->
</saml:Attribute>
<saml:Attribute Name="FirstPage">
  <saml:AttributeValue>pssssearch.aspx</saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="UpdateContact"> <!-- Mandatory -->
  <saml:AttributeValue>>false</saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="TrueLoginPage">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="ErrorLoginPage">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="RequestId">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="DvcSubmitFinal">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="DvcNoSubmitFinal">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
```

```
<saml:Attribute Name="PayEndDate">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="PayFREQCD">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="Erid">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="FirstName">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="LastName">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="UserID">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="Password">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="EmailAddress">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="AllPlanAccessCd">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="AllPlanAccessSecProfileId">
  <saml:AttributeValue></saml:AttributeValue>
```

```

</saml:Attribute>
<saml:Attribute Name="AccessbyPlan">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
<saml:Attribute Name="PlanID">
  <saml:AttributeValue></saml:AttributeValue>
</saml:Attribute>
</saml:AttributeStatement>
</saml:Assertion>
</samlp:Response>

```

Participant SAML SSO Request Mapping:

Code snippet

```

If Request("decryptassertions") IsNot Nothing
Then m_decryptAssertions =
Request("decryptassertions") End If
If Request("validateinboundtokenexpiration") IsNot
Nothing Then validateTokenExpiration =
Request("validateinboundtokenexpiration") End If
If Not String.IsNullOrEmpty(m_SAMLSP) or Not
String.IsNullOrEmpty(m_APID) Then Dim objBusiness As Relius.Admin.Web.Bus.clsSAML = New
Bus.clsSAML(objCommon) objBusiness.GetInBoundSAMLConfigurationData(m_SAMLSP, m_APID)
m_debugPath = objBusiness.GetDebugPath() m_debugMode = objBusiness.GetDebugMode()
Dim issuer As String = String.Empty

Dim timeExpiration As Integer = 0

Dim samlUtil As SamlUtility = New SamlUtility(m_debugPath, m_debugMode)

If Not String.IsNullOrEmpty(m_SAMLSP) Then samlUtil.GetClientSAMLConfiguration(m_SAMLSP,
objCommon.g_xmlStateDoc, m_RelayState, m_SAMLSubject, m_encryptAssertions, m_decryptAssertions,
m_clientAssertions, m_absoluteURL, timeExpiration, issuer, validateTokenExpiration) End If
m_x509Certificate_public = samlUtil.CreateX509Certificate(objBusiness.GetClientPublicX509Certificate,
String.Empty, True) m_x509Certificate =
samlUtil.CreateX509Certificate(objBusiness.GetReliusPrivateX509Certificate,

```

©2018 FIS and/or its subsidiaries. All Rights Reserved.

```

objBusiness.GetReliusPrivateCertificatePassword, False)

Dim internalTesting As Boolean = False

If (Request.QueryString.Keys.Count > 0 AndAlso Request.QueryString("internaltesting") = "true") Then If
(Request.QueryString("inbound") <> "true") Then internalTesting = True End If m_x509Certificate =
samlUtil.CreateX509Certificate(AppDomain.CurrentDomain.BaseDirectory + "\ssolidp.cer", String.Empty, True) '
m_decryptAssertions = "false" End If

Dim saml2 As clsSAML2 = New clsSAML2(m_debugPath, m_debugMode)
saml2.ProcessSAMLToken(Request, Response, m_x509Certificate_public, m_x509Certificate,
m_decryptAssertions, validateTokenExpiration, internalTesting) End If

////////////////////////////////////

If m_ReceivedAssertions IsNot Nothing AndAlso m_ReceivedAssertions.Count > 0 Then

For Each tempAssertion As KeyValuePair(Of String, String) In m_ReceivedAssertions If
String.Compare(tempAssertion.Key, "SSN", True) = 0 Then

'ssn =
tempAssertion.Value ssn

=

Relius.Admin.Web.Utilities.clsEncrypt.ConvertToHex(Relius.Admin.Web.Utilities.clsEncrypt.SymmetricEncrypt
String(tempAssertion.Value, "@ReLiUsKeY@"))

Elseif String.Compare(tempAssertion.Key, "FIRSTPAGE", True)
= 0 Then firstpage = tempAssertion.Value

Elseif String.Compare(tempAssertion.Key, "PLANID", True)
= 0 Then planId = tempAssertion.Value

Elseif String.Compare(tempAssertion.Key, "TRUELOGINPAGE", True) = 0
Then trueLoginPage = tempAssertion.Value

Elseif String.Compare(tempAssertion.Key, "ERRORLOGINPAGE", True) = 0
Then errorLoginPage = tempAssertion.Value

Elseif String.Compare(tempAssertion.Key, "WEBCOOKIE", True) =
0 Then webCookie = tempAssertion.Value

Elseif String.Compare(tempAssertion.Key, "LANGUAGE", True) =

```

©2018 FIS and/or its subsidiaries. All Rights Reserved.


```

If Request("decryptassertions") IsNot Nothing Then
    m_decryptAssertions = Request("decryptassertions")
End If

If Request("validateinboundtokenexpiration") IsNot Nothing Then
    validateTokenExpiration = Request("validateinboundtokenexpiration")
End If

If Not String.IsNullOrEmpty(m_SAMLSP) or Not String.IsNullOrEmpty(m_APID) Then
    Dim objBusiness As Relius.Admin.Web.Bus.clsSAML = New Bus.clsSAML(objCommon)
    objBusiness.GetInboundSAMLConfigurationData(m_SAMLSP, m_APID)
    m_debugPath = objBusiness.GetDebugPath()
    m_debugMode = objBusiness.GetDebugMode()

    Dim issuer As String = String.Empty
    Dim timeExpiration As Integer = 0

    Dim samlUtil As SamlUtility = New SamlUtility(m_debugPath, m_debugMode)

    If Not String.IsNullOrEmpty(m_SAMLSP) Then
        samlUtil.GetClientSAMLConfiguration(m_SAMLSP, objCommon.g_xmlStateDoc, m_RelayState, m_SAMLSubject,
            m_encryptAssertions, m_decryptAssertions, m_clientAssertions, m_absoluteURL, timeExpiration, issuer, validateTokenExpiration)
    End If
    m_x509Certificate_public = samlUtil.CreateX509Certificate(objBusiness.GetClientPublicX509Certificate, String.Empty, True)
    m_x509Certificate = samlUtil.CreateX509Certificate(objBusiness.GetReliusPrivateX509Certificate, objBusiness.GetReliusPrivateCertificatePassword, False)

    Dim internalTesting As Boolean = False
    If (Request.QueryString.Keys.Count > 0 AndAlso Request.QueryString("internaltesting") = "true") Then
        If (Request.QueryString("inbound") <> "true") Then
            internalTesting = True
        End If
        m_x509Certificate = samlUtil.CreateX509Certificate(AppDomain.CurrentDomain.BaseDirectory + "\sso\idp.cer", String.Empty, True)
        m_decryptAssertions = "false"
    End If

    Dim saml2 As clsSAML2 = New clsSAML2(m_debugPath, m_debugMode)
    saml2.ProcessSAMLToken(Request, Response, m_x509Certificate_public, m_x509Certificate, m_decryptAssertions, validateTokenExpiration, internalTesting)
End If

```



```

If m_ReceivedAssertions IsNot Nothing AndAlso m_ReceivedAssertions.Count > 0 Then
    For Each tempAssertion As KeyValuePair(Of String, String) In m_ReceivedAssertions
        If String.Compare(tempAssertion.Key, "SSN", True) = 0 Then
            'ssn = tempAssertion.Value
            ssn = Relius.Admin.Web.Utilities.clsEncrypt.ConvertToHex(Relius.Admin.Web.Utilities.clsEncrypt.SymmetricEncrypt
        ElseIf String.Compare(tempAssertion.Key, "FIRSTPAGE", True) = 0 Then
            firstpage = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "PLANID", True) = 0 Then
            planId = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "TRUELOGINPAGE", True) = 0 Then
            trueLoginPage = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "ERRORLOGINPAGE", True) = 0 Then
            errorLoginPage = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "WEBCOOKIE", True) = 0 Then
            webCookie = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "LANGUAGE", True) = 0 Then
            language = tempAssertion.Value
        End If
        If m_printOnScreen Then
            m_httpResponse.Write("<b>" + tempAssertion.Key + "</b> &nbsp;&nbsp; " + tempAssertion.Value + "<br/>")
        End If
    Next
End If
WriteSamLog("Fill client assertion to post completed", Severity.None)
WriteSamLog("Post data to participant SSO started", Severity.None)
If Not m_printOnScreen Then
    If (Not String.IsNullOrEmpty(ssn)) Then
        Dim partHtmlWeb As StringBuilder = New StringBuilder()
        partHtmlWeb.Append("<html><body><form name='samlss0' action='participantsinglesignon.aspx' method='post'>")
        partHtmlWeb.Append("<input type='hidden' name='SSNUM' value='" + ssn + "'/>")
        partHtmlWeb.Append("<input type='hidden' name='FIRSTPAGE' value='" + firstpage + "'/>")
        partHtmlWeb.Append("<input type='hidden' name='PLANID' value='" + planId + "'/>")

        partHtmlWeb.Append("<input type='hidden' name='ISMAMLSO' value='True'/>")
        partHtmlWeb.Append("<input type='hidden' name='TRUELOGINPAGE' value='" + trueLoginPage + "'/>")
        partHtmlWeb.Append("<input type='hidden' name='ERRORLOGINPAGE' value='" + errorLoginPage + "'/>")
        partHtmlWeb.Append("<input type='hidden' name='WEBCOOKIE' value='" + webCookie + "'/>")
        partHtmlWeb.Append("<input type='hidden' name='LANGUAGE' value='" + language + "'/>")

        partHtmlWeb.Append("</form></body><script type='text/javascript'>document.samlss0.submit();</script></html>")
        m_httpResponse.Write(partHtmlWeb.ToString())
    End If
End If

```

Sponsor SAML SSO Request Mapping:

```

If Request("decryptassertions") IsNot Nothing Then
    m_decryptAssertions = Request("decryptassertions")
End If

If Request("validateinboundtokenexpiration") IsNot Nothing Then
    validateTokenExpiration = Request("validateinboundtokenexpiration")
End If

If String.IsNullOrWhiteSpace(m_SAMLSP) And String.IsNullOrWhiteSpace(m_APID) Then
    m_errorMessage = m_ErrorMsg_ClientIdMissing
    m_Error = True
    RedirectToErrorPage()
End If

Dim l_clsSAML As clsSAML = New clsSAML(objCommon)
m_debugPath = l_clsSAML.GetDebugPath()
m_debugMode = l_clsSAML.GetDebugMode()
If String.IsNullOrWhiteSpace(m_debugMode) Then
    m_debugMode = False
End If
m_Saml2 = New clsSAML2(m_debugPath, m_debugMode)
m_Saml2.WriteSanLog("SAML SSO Started", Logging.Severity.None)
If Request.Form IsNot Nothing AndAlso Request.Form.Keys IsNot Nothing AndAlso Request.Form.Keys.Count > 0 Then
    For Each key As String In Request.Form.Keys
        m_Saml2.WriteSanLog("<br>" + Request.Form(key), Logging.Severity.None)
    Next
End If
l_clsSAML.GetInBoundSAMLConfigurationData(m_SAMLSP, m_APID)

Dim issuer As String = String.Empty
Dim timeExpiration As Integer = 0

Dim sanUtil As SanUtility = New SanUtility(m_debugPath, m_debugMode)
Dim l_isSAMLValid As Boolean = False

If Not String.IsNullOrWhiteSpace(m_SAMLSP) Then
    sanUtil.GetClientSAMLConfiguration(m_SAMLSP, Nothing, m_RelayState, m_SAMLSubject, m_encryptAssertions, m_decryptAssertions, m_clientAssertions, m_absoluteURL, timeExpiration, is
End If

m_x509Certificate_public = sanUtil.CreateX509Certificate(l_clsSAML.GetClientPublicX509Certificate, String.Empty, True)
m_x509Certificate = sanUtil.CreateX509Certificate(l_clsSAML.GetReliusPrivateX509Certificate, l_clsSAML.GetReliusPrivateCertificatePassword, False)

'Process SAML Token which is received from client
If m_x509Certificate_public Is Nothing Then
    m_Error = True
    m_errorMessage = m_ErrorMsg_MissingClientPublicCertificate
End If

If (m_Error = False AndAlso String.Compare(m_decryptAssertions, "true", True) = 0 AndAlso m_x509Certificate Is Nothing) Then
    m_Error = True
    m_errorMessage = m_ErrorMsg_MissingReliusPrivateCertificate
End If

If m_Error = False Then
    If m_Saml2.ProcessSAMLToken(Request, Response, m_x509Certificate_public, m_x509Certificate, m_decryptAssertions, m_receivedClientAssertions, validateTokenExpiration) Then
        l_isSAMLValid = True
    End If

```

```

IT m_receivedClientAssertions ISNOT NOTHING AND ALSO m_receivedClientAssertions.COUNT > 0 THEN
    m_clsSAMLContact = New clsSAMLContact()
    For Each tempAssertion As KeyValuePair(Of String, String) In m_receivedClientAssertions
        If String.Compare(tempAssertion.Key, "APIID", True) = 0 Then 'APIID
            m_clsSAMLContact.APIID = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "CONTACTID", True) = 0 Then 'Contact ID
            If (Not String.IsNullOrEmpty(tempAssertion.Value)) Then
                m_clsSAMLContact.ContactId = tempAssertion.Value.Trim()
            End If
        ElseIf String.Compare(tempAssertion.Key, "FIRSTPAGE", True) = 0 Then 'First Page
            m_clsSAMLContact.FirstPage = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "TRUELOGINPAGE", True) = 0 Then 'True Login Page
            m_clsSAMLContact.TrueLoginPage = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "ERRORLOGINPAGE", True) = 0 Then 'Error Login Page
            m_clsSAMLContact.ErrorLoginPage = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "PLANID", True) = 0 Then 'Plan ID
            m_clsSAMLContact.PlanId = tempAssertion.Value

        ElseIf String.Compare(tempAssertion.Key, "REQUESTID", True) = 0 Then 'REQUEST ID
            m_clsSAMLContact.RequestId = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "DVCSUBMITFINAL", True) = 0 Then 'DVC SUBMIT FINAL
            m_clsSAMLContact.DvcSubmitFinal = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "DVCNOSUBMITFINAL", True) = 0 Then 'DVC NO SUBMIT FINAL
            m_clsSAMLContact.DvcNoSubmitFinal = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "PAYENDDATE", True) = 0 Then 'PAY END DATE
            m_clsSAMLContact.PayEndDate = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "PAYFREQC", True) = 0 Then 'PAY FREQC
            m_clsSAMLContact.PayFREQC = tempAssertion.Value

        ElseIf String.Compare(tempAssertion.Key, "LOGINTYPE", True) = 0 Then 'LOGIN TYPE
            m_clsSAMLContact.LoginType = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "SSN", True) = 0 Then 'SSN
            m_clsSAMLContact.SSNum = tempAssertion.Value

        ElseIf String.Compare(tempAssertion.Key, "UPDATECONTACT", True) = 0 Then 'Update Contact
            If (Not String.IsNullOrEmpty(tempAssertion.Value)) Then
                m_clsSAMLContact.UpdateContact = String.Compare(tempAssertion.Value, "TRUE", True) = 0
            End If
        ElseIf String.Compare(tempAssertion.Key, "ERID", True) = 0 Then
            m_clsSAMLContact.ErID = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "FIRSTNAME", True) = 0 Then
            m_clsSAMLContact.FirstName = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "LASTNAME", True) = 0 Then
            m_clsSAMLContact.LastName = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "USERID", True) = 0 Then
            m_clsSAMLContact.UserId = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "PASSWORD", True) = 0 Then
            m_clsSAMLContact.Password = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "EMAILADDRESS", True) = 0 Then
            m_clsSAMLContact.EmailAddress = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "ALLPLANACCESSCD", True) = 0 Then
            If (Not String.IsNullOrEmpty(tempAssertion.Value)) Then
                m_clsSAMLContact.AllPlanAccessCd = String.Compare(tempAssertion.Value, "true", True) = 0
            End If
        ElseIf String.Compare(tempAssertion.Key, "ALLPLANACCESSSECPROFILEID", True) = 0 Then
            m_clsSAMLContact.AllPlanAccessSecProfileId = tempAssertion.Value
        ElseIf String.Compare(tempAssertion.Key, "ACCESSBYPLAN", True) = 0 Then
            m_clsSAMLContact.AccessByPlanXmlNode = tempAssertion.Value
    End For

```


For Internal testing (Note the Client Private certificate and Relius public key must reside in the \reliusweb\SSO of the web server to use this test page).

You can test by using testpage as mention below:

Server/Website Name/testsaml2.aspx

← → ↻ ⓘ localhost/ReliusWeb2017.2/testsaml2.aspx

Please Select **User Type**

Please Select **InBound/OutBound SSO:**

Please Select **Client:**

For Participant Inbound:

← → ↻ ⓘ localhost/ReliusWeb2017.2/testsaml2.aspx

Please Select **User Type**

Please Select **InBound/OutBound SSO:**

Please Select **Client:**

Generic Inbound - Relius Web Participant

SSN: *

PlanId:

First Page:

AP ID:

SAML URL: Enter full site URL if SAML token is to be posted to different site.

True Login Page:

Error Login Page:

WEB COOKIE:

LOGINTYPE:

Please Select language:

For Sponsor Inbound:

localhost/ReliusWeb2017.2/testsaml2.aspx

Please Select **User Type**:

Please Select **InBound/OutBound SSO**:

Please Select **Client**:

Generic Sponsor SAML SSO

APID *:
 ContactID *:
 SSN:
 Please Select Login Type:
 First Page:
 True Login Page:
 Error Login Page:
 Plan ID:
 Update Contact: (true/false) **Note:** Below contact details will update/insert only if **Update Contact** set to 'true'
 Request ID:
 DVC Submit Final:
 DVC No Submit Final:
 Pay End Date:
 Pay FREQCD:

Contact Details

EriId:
 First Name:
 Last Name:
 Email Address:
 User ID:
 Password:
 All Plan Access Cd: (true/false)
 All Plan Access Security Profile Id: For Relius_View: Constant value of "11", for Relius_Modify: "Constant Value of "1"
 Access by Plan: Format: PLANID;SECURITYPROFILEID;PLANID;SECURITYPROFILEID Example: 123;456;789;000

SAML URL and Cert Password

SAML URL: Enter full site URL if SAML token is to be posted to different site.
 Client Certificate Password:

Certificate Generation Utility

1. Open <https://www.relius.net/Support/Technology.aspx> in your browser

- Click on “Self Signed RSA 256 Bit X509 Certificate Generation Utility” in the Request Installation Media and Special Services Section

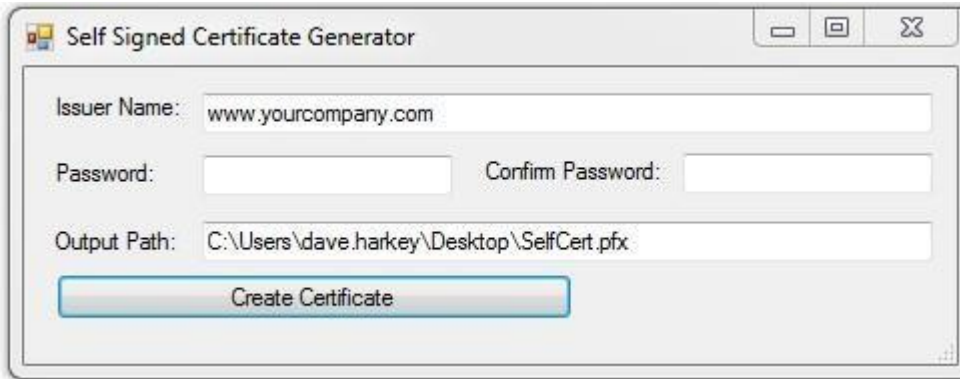
The screenshot shows the FIS Relius website interface. The top navigation bar includes links for 'About Us', 'Support', 'Products and Services', 'News', 'Training', and 'Events'. The main content area is divided into several sections:

- Relius Software:** Documents PC, Documents ASP, Administration, STP, Proposal, Education, Government Forms, Connect, Technology.
- Support Services:** Document Consulting, Client Services, Relius Newsletter, PPA Restatement, RSA Device Assignment, Request to Cancel Product.
- Applications:** Relius Documents, Subscribe.
- Relius Technology (Administration, Documents, Proposal & Forms):**
 - Technology News:** News, Updates and Important Information; Relius Supported Operating Systems; FIS Relius ASP SMTP Mail Routing Information.
 - Installation and Upgrade Documents:** Current installation and upgrade documents by version.
 - Hardware and Upgrade Requirements:** Administration 20.x Multi-user (Network) Requirements; Administration 20.x Single-user (Standalone) Requirements; Administration 19.x Multi-user (Network) Requirements; Administration 19.x Single-user (Standalone) Requirements; Administration 18.x Multi-user (Network) Requirements; Administration 18.x Single-user (Standalone) Requirements; Proposal 10.0 Hardware Requirements; Proposal 9.0 Hardware Requirements; Government Forms Hardware Requirements; Administration Calculate Oracle Database Server Recommendation.
 - Request Installation Media And Special Services:** Administration Submit Oracle Installation Media Request; Request On-Site Installation / Upgrade; Request priority or after-hours (evening or weekend) telephone support; Personal Oracle 11 (Standalone Databases only) installation DVD (3 GB); **Self Signed RSA 256 Bit X509 Certificate Generation Utility** (highlighted with a red arrow).
 - ASP (Application Service Provider), RRS (Relius Recovery Services), and SuiteManager:** Suite Manager 3.1.1 and Installation Instructions; Suite Manager 4.0 (for RRS and Oracle 11g Servers only).
 - Environment O/S Patch Certification:** Certified O/S Patches.

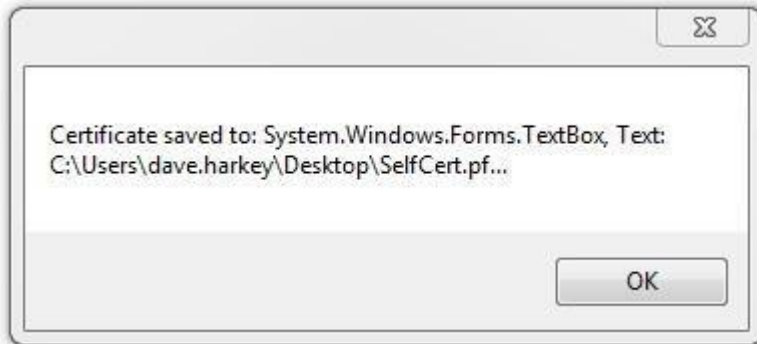
On the right side, there is a promotional banner for the 'CHICAGO ADVANCED PENSION CONFERENCE' held from September 5-8, 2017, with a follow-up event on 2/7-9/2018 in Orlando, FL.

© 2017 FIS. Empowering the Financial World™. Home | An FIS Company | Legal Stuff | Continuity Statement

3. Navigate to the Downloads folder on your PC
4. Open the Relius.SelfCert.exe ZIP file using a zip tool (such as WinZip or 7-Zip)
5. Double Click on the Relius.SelfCert.exe file. The following screen will be displayed:



6. Issuer Name: enter the URL for your Relius Web Application
7. Password: Enter the password you are assigning to the certificate
8. Confirm Password: Re-enter the password you are assigning to the certificate
9. Output Path: Enter the location where you would like the certificate saved to (keep in mind that you can create on your PC and then copy/move it to your webserver)
10. Click Create Certificate. The box below will be displayed:

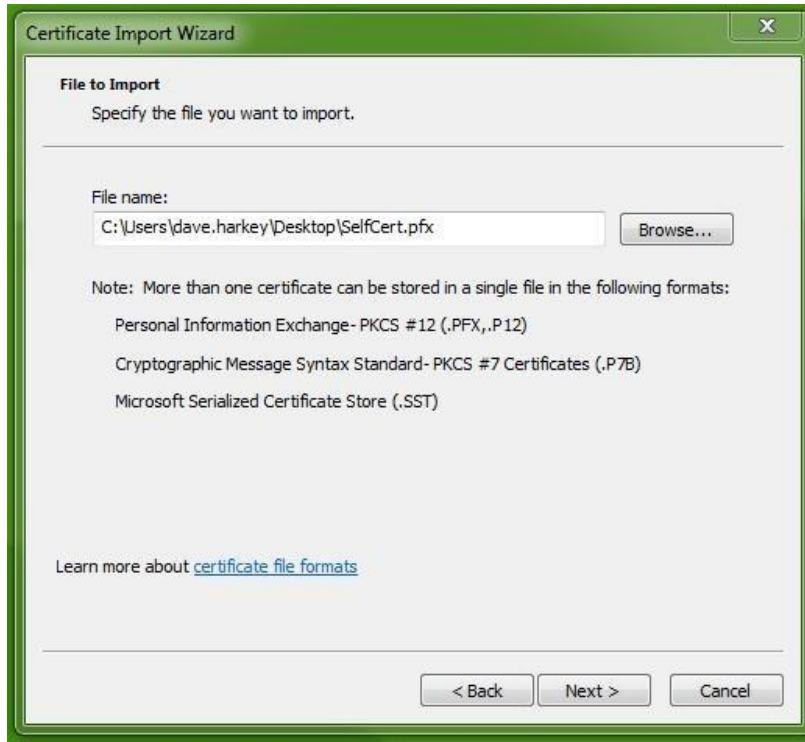


11. Click ok
12. Close the Certificate Generator
13. Navigate to the directory where you saved the certificate

14. Double Click on the SelfCert file to open up the Certificate Import Wizard. The following screen will be displayed:



15. Click Next. The following screen will appear with the file name prefilled with the location of the certificate file



16. Click Next. The following screen will be displayed:



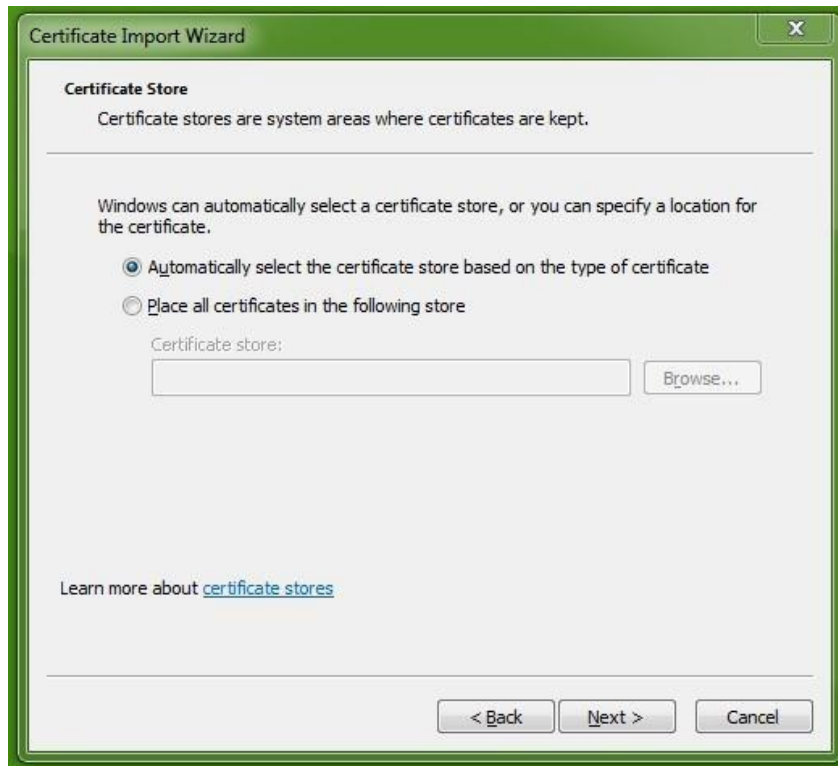
The image shows a screenshot of the 'Certificate Import Wizard' dialog box. The title bar reads 'Certificate Import Wizard' with a close button (X) on the right. The main content area is titled 'Password' and contains the following text: 'To maintain security, the private key was protected with a password.' Below this is a horizontal line, followed by the instruction 'Type the password for the private key.' and a label 'Password:' above a text input field. There are three checkboxes: the first is 'Enable strong private key protection. You will be prompted every time the private key is used by an application if you enable this option.' (unchecked); the second is 'Mark this key as exportable. This will allow you to back up or transport your keys at a later time.' (unchecked); and the third is 'Include all extended properties.' (checked). At the bottom left, there is a blue hyperlink: 'Learn more about [protecting private keys](#)'. At the bottom right, there are three buttons: '< Back', 'Next >', and 'Cancel'.

17. Enter the password for the certificate (the password from step 7)

18. Click on the middle box to check it.

19. Leave the last box checked

20. Click Next. The following screen will be displayed:



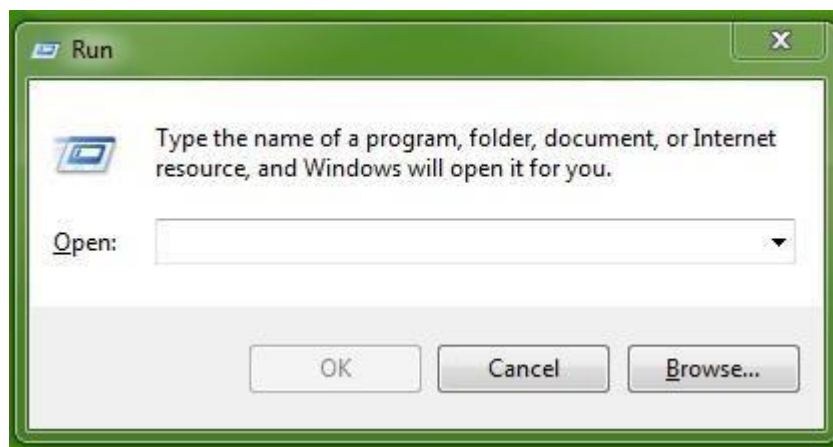
21. Click on the radio button as shown above

22. Click next for the following screen

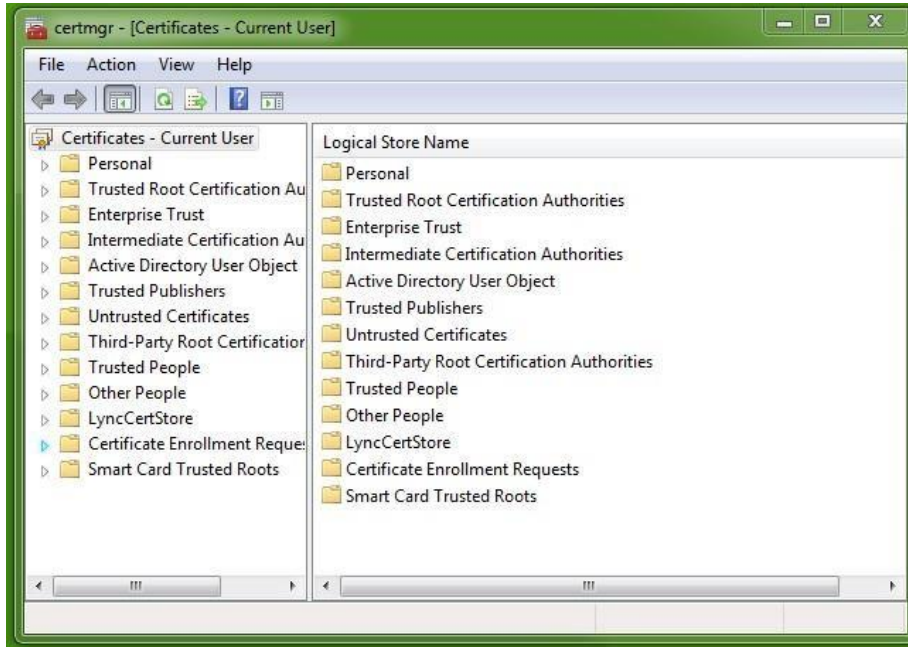


23. Click Finish

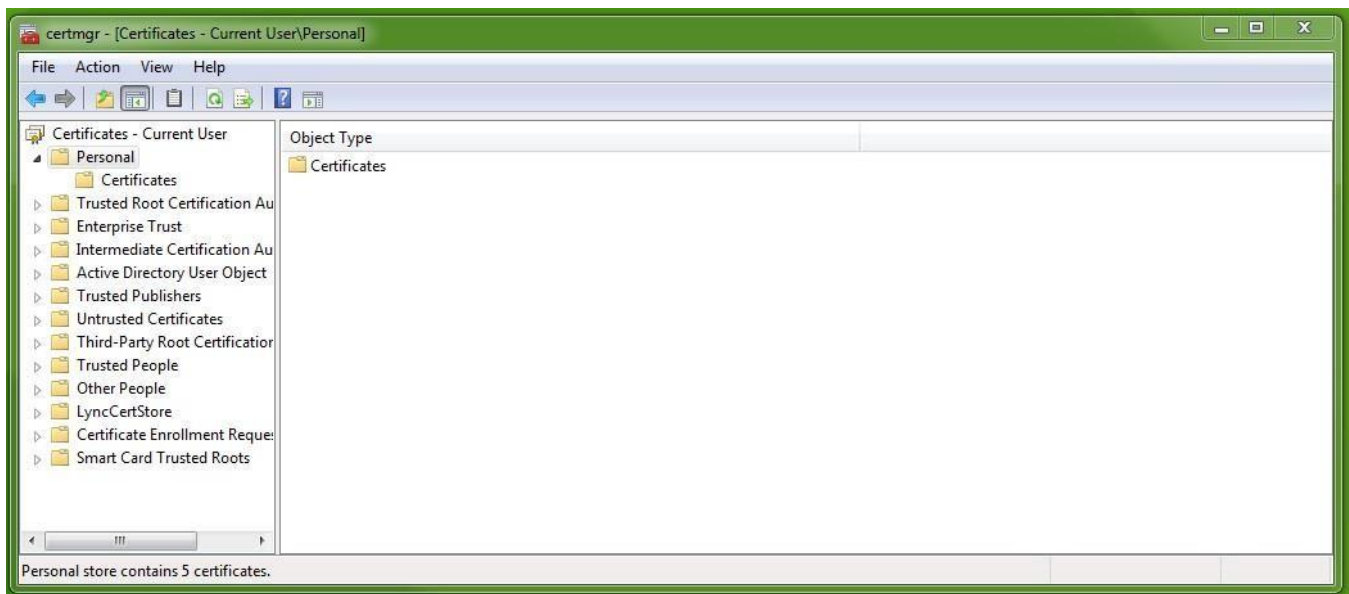
24. Open the Windows Run Program (this can be found in your Accessories programs or by pressing the windows key and R or by clicking on the window icon and typing run in the search programs and files box). The following screen will be displayed:



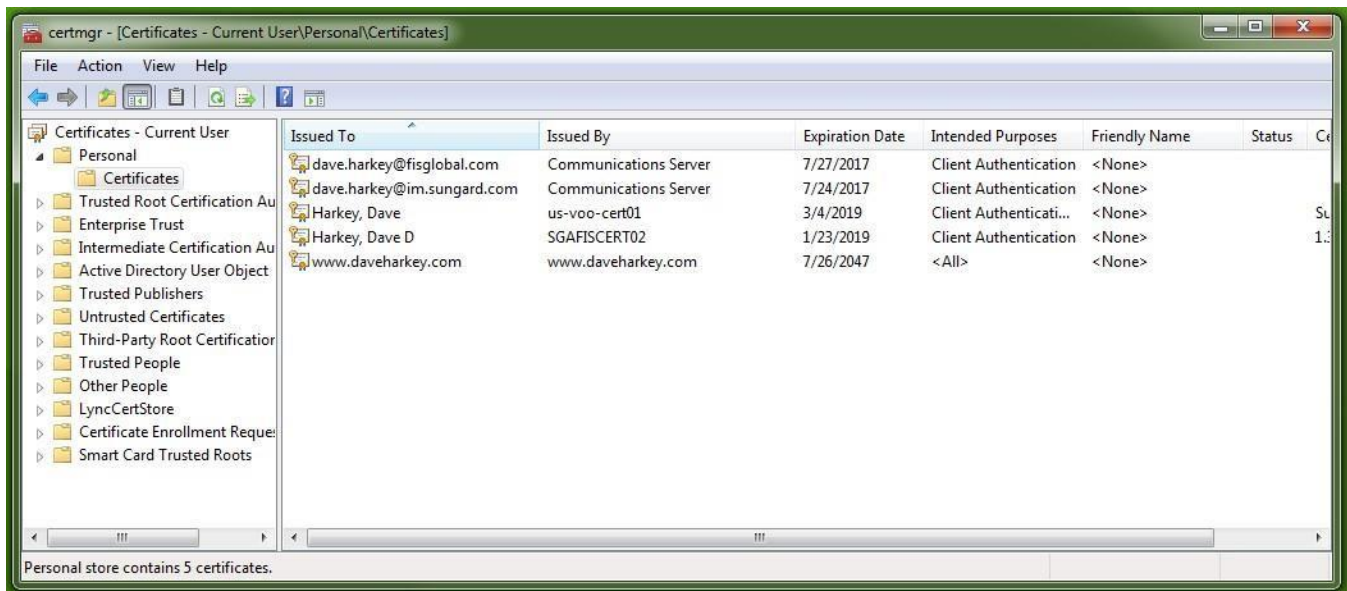
25. Enter certmgr.msc on the open Box and click ok. After a short wait the following screen will be displayed:



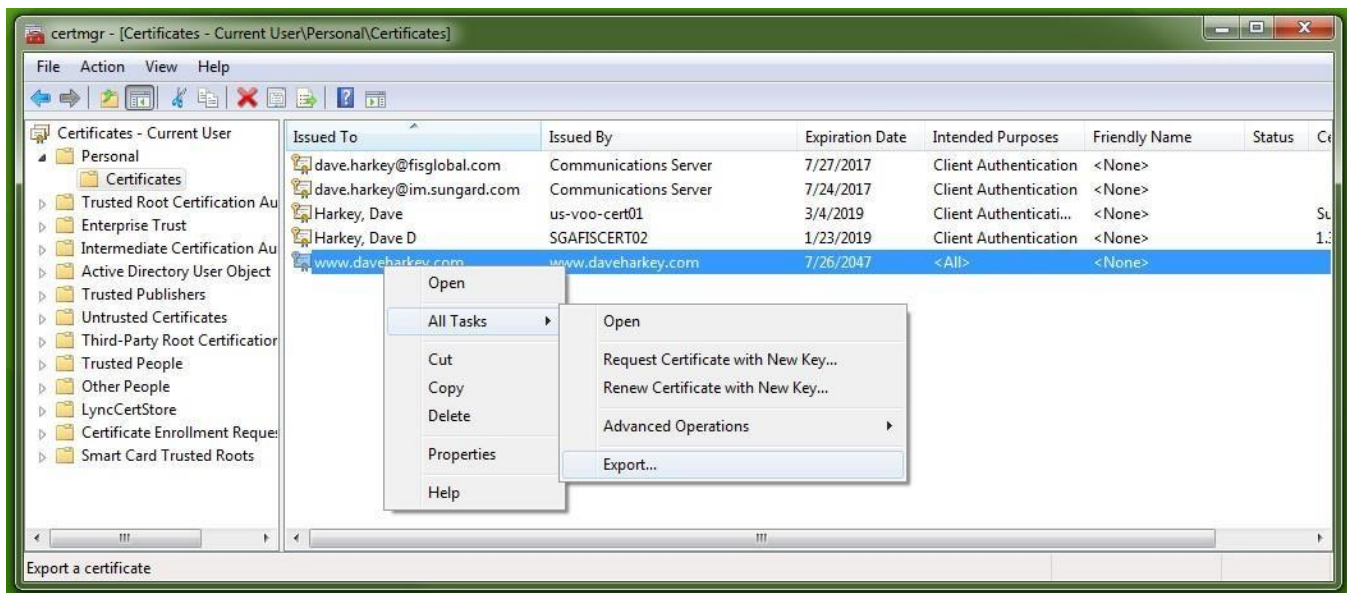
26. Double Click on the Personal Folder in the Logical Store Name Box to get the following screen:



27. Double Click on the Certificates Folder in the Object Type Box to get the following screen:



28. Right Click on the certificate with the same name as the Issuer Name in Step 6



29. Click on All Task

30. Click on Export to get the following screen:



31. Click Next



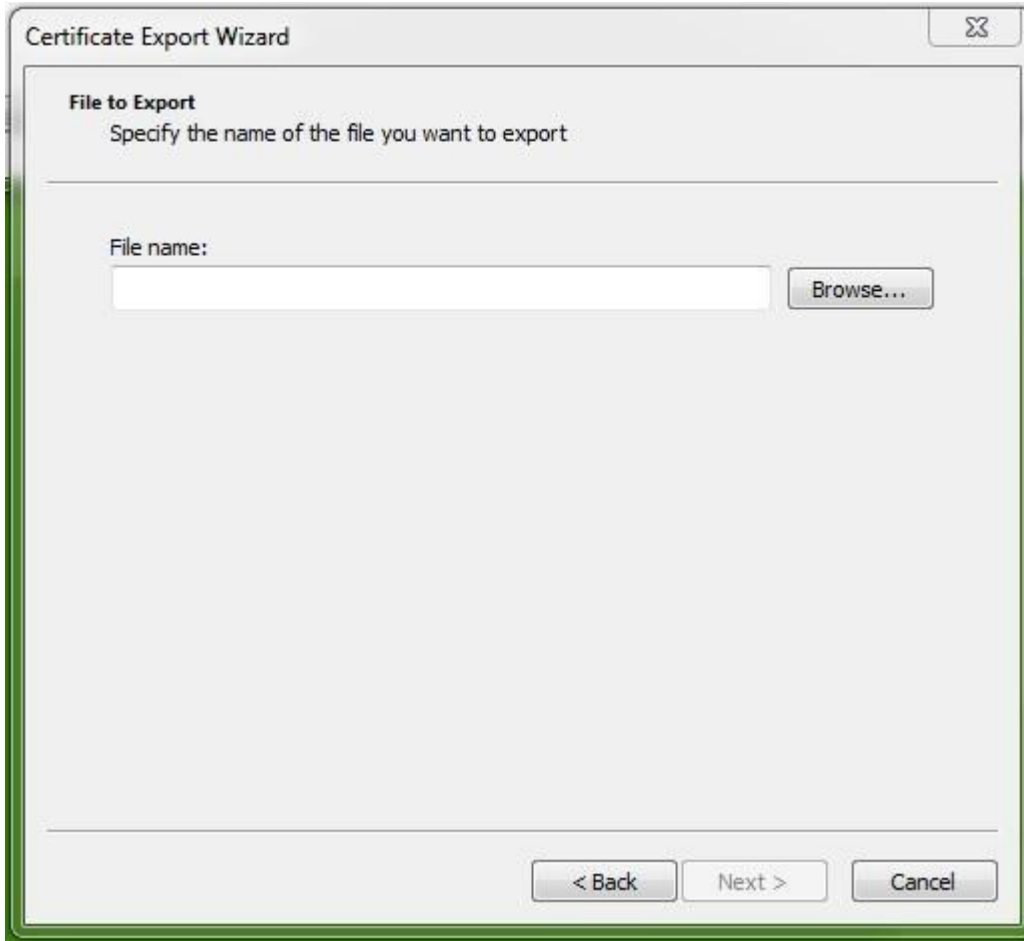
32. Click on the Radio Button as shown above

33. Click Next



34. Click on the Radio Button as shown above

35. Click Next



36. Enter the name that you want for you certificate 37. Click Next



38. Make note of the location where the file will be placed across from File Name above – this is where you will find the file
39. Click Finish – you should see the following:



40. Click ok

41. Close the certmgr screen
42. Update the TPA information in Relius with the new PFX file created by the Certificate Generator.
43. Repeat for Client certificate and update the .cer file in Relius.

Single Sign-On Legacy System Overview

Note: If you have set up SAML SSO and want to implement Legacy SSO you must first remove the certificate you setup in the Modify Advice/Provider screen. This will allow you to create the legacy adviceprovider client config file in Web Configuration using the Generic Provider. This config file will allow you to generate a token for legacy SSO.

Legacy SSO allows client applications to access the Relius Web application via the Single Sign-On service. A client application will request a token from the Relius SSO tokengenerator service. The tokengenerator service will then associate that token to the user attempting to log onto Relius Web. Once the token is returned, the client application will then forward the resulting token to Relius via an HTTP or HTTPS POST request to the appropriate SSO aspx page (participant/sponsor/advisor). Upon successful validation of the token by ReliusWeb, the participant/sponsor/advisor will be logged in to Relius. **The token generation itself is application-neutral and is only there to create the token and associate it to the User ID that is passed in.**

Legacy Single Sign On Steps

The SSO web service is included as part of the Relius Web Solution application. The sequence diagram below illustrates how the SSO will function for Relius Web Solution.

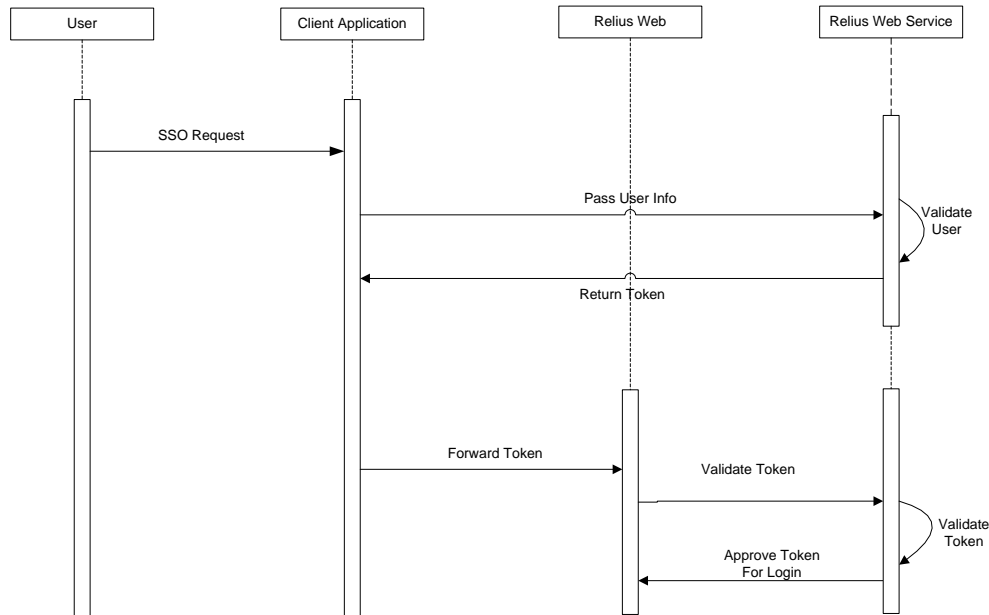


Figure 1: Inbound Single Sign-on Sequence Diagram

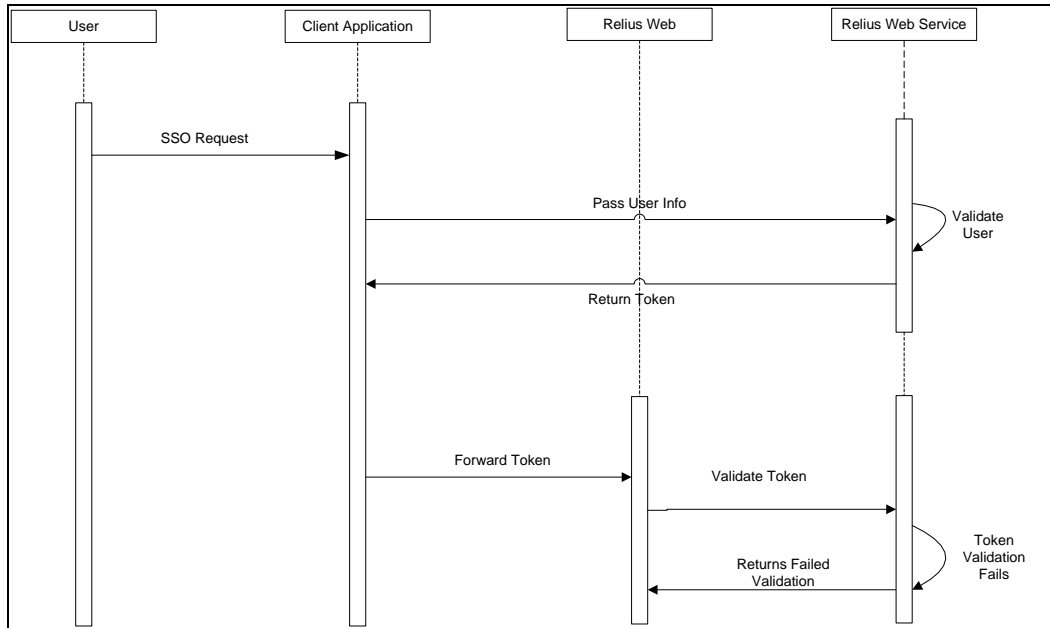
1. A user successfully logs into a client's site.
2. From the client's site, the user makes a request to visit the associated ReliusWeb product (participant/sponsor/advisor).
3. The client site forwards necessary credentials (ssn or webcookie for participants – or contact ID for plan sponsors and/or advisors) to the Relius SSO tokengenerator service using a server-to-server connection.
4. The web service creates a token and time stamp and then stores the token and timestamp in the database to associate with the user. **The time-stamped token is valid for 1 minute.**
5. The web service will then return the token to the client site.
6. The client forwards the received token to ReliusWeb via an HTTP or HTTPS POST with the token embedded in the post.
7. ReliusWeb page consumes the token to the internal SSO service for validation.
8. The SSO service validates the token and time stamp against the matching user information in the database and passes the encrypted user information to the ReliusWeb page.
9. ReliusWeb then decrypts the returned string to derive the participant ID or Contact ID.
10. ReliusWeb allows user access, bypassing the standard login procedure. The user is initially presented the Account Summary page.

Error Handling

Exceptions to the process may arise when validating the user or the generated token, or if the user's session is allowed to remain idle and expire after a successful log in.

Token Fails Validation

If the SSO web service indicates that the generated token cannot be validated for a valid user, the user is directed to the standard Relius login page or the alternative error page defined by the client in the `<ERRORLOGINPAGE>` value passed in through the form post.



Session Expiration

If the user's session is allowed to remain idle and expire after a successful single sign-on, then Relius will end the user's session and direct the user to the `<TRUELOGINPAGE>` value identified in the form post. Depending how the system is setup, the user will either be required to access the client's site and go through the single sign-on process again to obtain access to Relius, or the user may have the option to use a Relius login ID and password to log directly back into the ReliusWeb page.

Communication Protocols

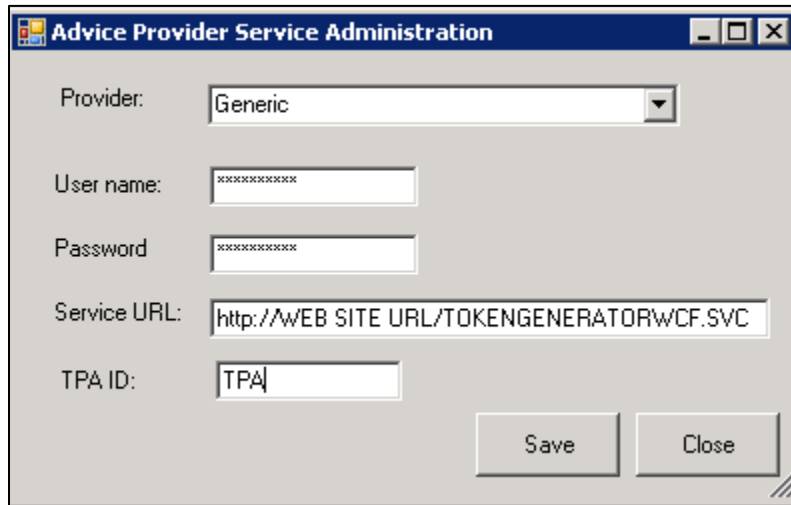
The following protocols are in place to allow Legacy SSO to operate seamlessly for users.

Token Creation

In order to use the web services a set of two configuration files (.config) are needed. The configuration files are used as matching files. One is used to assist the caller of the web service for identifying the userID and password. The other file will remain on the web server and is used to help the web service authenticate the user that is calling the services.

1. From the ReliusWeb server, launch the the Web Configuration Utility (*Start | Programs | Relius | Admin | Web Configuration*).
2. Select the site that will be utilizing the SSO service.

3. On the main menu, click *Web Site | Set Advice Provider Service*. This option will open a screen called Advice Provider Service Administration.
4. Select the 'Generic' option under Provider.
5. Enter a user name and password that you want to use for the connection to the web service. **These are user-defined values of your own creation.**
6. Modify the service URL to be your website URL with `TokenGeneratorWcf.svc` appended to it. Example: `https://www.websiteurl.com/ReliusWeb/TokenGeneratorWcf.svc`
7. Create a TPAID for the service connection. **This is a user defined value of your own creation.**



8. Click **Save** to generate the configuration files. The files will be saved in the `C:\Program Files (x86)\Relius\Admin\AUTHORIZATION_FILES` folder.
 - a. `Adviceproviderclient*.config` is to be sent to the connecting party. This file is the one that you will need to reference when the web service is called so you can find that file and forward it to your development area so they can use it for connecting to the service.
 - b. `Adviceproviderserver*.config` should remain in the folder in which it was created. This is the configuration file is used by the service to help authenticate the user.

Token Generation

The token generation process requires a client call to the token generation web service `tokengeneration.asmx`. The call to this web service will include the value of the SSN or web cookie for a participant – or the ContactID for a sponsor/advisor. The identification values will be included in the `<STOREDVALUE>` tag in the post. The token generation process does not validate the user; it only creates a token associated with the ID that is passed in.

The `<ADVICEPROVIDERCD>` and `<TPAID>` values needed for the connection can be found in the file name of the `AdviceProviderClient` configuration file that you will use for the connection. For example, a file name of `adviceproviderclient_1_client1.config` has an `AdviceProviderCd = 1` and a `TPAID = client1`.

The `<USERNAME>` value and the `<PASSWORD>` value can be found under the same names within the XML string in the configuration file.

The URL of the web service can be used by replacing the .asmx name in the config file for the <SERVICEURL> with tokengenerator.asmx.

Sample **Post to** the tokengenerator.asmx:

```
<?xml version="1.0" encoding="utf-8"?><soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><soap:Header><AuthenticationHeader
xmlns="http://tempuri.org/"><UserName>STRING</UserName><Password>STRING</Password>
<TPAID>STRING</TPAID><AdviceProviderCd>STRING</AdviceProviderCd><AgentID
/></AuthenticationHeader></soap:Header><soap:Body><CreateToken
xmlns="http://tempuri.org/"><StoredValue>STRING</StoredValue></CreateToken></soap:Body></soap:Envelope>
```

Token Response

The response will include a <CREATETOKENRESULT> tag that will provide the token that is required to pass into the form post for the login. **The token is valid for 1 minute from the creation.**

Sample **Response from** the tokengenerator.asmx:

```
<?xml version="1.0" encoding="utf-8"?><soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><soap:Header><AuthenticationHeader
xmlns="http://tempuri.org/"><UserName>STRING</UserName><Password>STRING</Password>
<TPAID>STRING</TPAID><AdviceProviderCd>STRING</AdviceProviderCd><AgentID
/></AuthenticationHeader></soap:Header><soap:Body><CreateTokenResponse
xmlns="http://tempuri.org/"><CreateTokenResult>STRING</CreateTokenResult></CreateTokenResponse></soap:Body></soap:Envelope>
```

Client to SSO Web Service

The communication between client and the Relius SSO web service will be across HTTPS and rely on **WCF wsHttpBinding Transport Security**. The participant or sponsor ID will be passed in the soap request.

Form Post to SSO page

After the token request, a form post will be made to the appropriate SSO page using at the client's URL. The page would be directed to one of the following:

- Participant web.....participantSingleSignon.aspx
- Sponsor web.....SponsorSingleSignon.aspx
- Advisor web.....AdvSingleSignon.aspx

Include the token value returned from the token request in the post.

There are some optional settings that can be passed in with the form post. We would recommend that you use the `<TRUELOGINPAGE>` and `<ERRORLOGINPAGE>` settings to control the flow of the user experience. These settings include:

<code><LANGUAGE></code>	Valid values are <code>ENG</code> or <code>SPA</code> to identify if the web should display the English or Spanish version of the web to the user.
<code><TRUELOGINPAGE></code>	Pass the Return URL to take the user when they log off of the Relius site or when their session times out.
<code><ERRORLOGINPAGE></code>	Pass a URL to take the user if the login process fails. This would be a page the client has created to handle any errors.

Note: For security purposes, Relius will not pass back much content in any error messages. If errors do occur, we recommend requesting assistance from Relius support to help debug any issues.

Additional settings are described later in this document. See the section called, "Optional Field Variables."

Sample HTTP POST Request Form

The following is an example of sample HTML that illustrates the use of the token returned from the SSO web service to single sign-on into the Relius Web application:

```
<form id="PostForm"
  name="PostForm"
  action="https://Reliuswebsolutions.com/Login/InboundSingleSignon/"
  method="POST"
  target="NewWin">
  <input type="hidden" name="SUTSTOKEN" value="STRING">
</form>

<script language='javascript'>
  var vPostForm = document.PostForm;
  window.open("", "NewWin", "");
  var w = window.setTimeout("vPostForm.submit();", 500);
</script>");
```

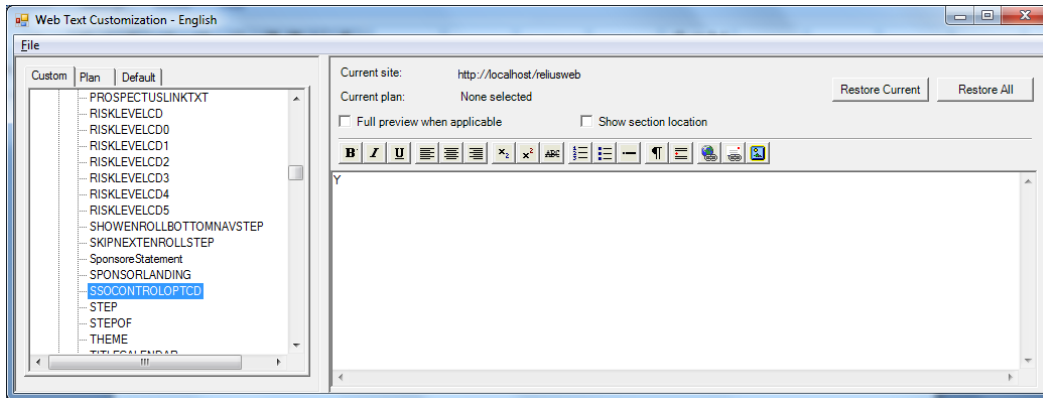
Settings for Legacy SSO

There are specific configuration settings that are required to enable SSO. This section provides detail regarding the implementation of the SSO feature of ReliusWeb Solutions. In addition, it identifies the configuration settings in the `websettings.config` file and resource strings files that need to be updated in order to utilize the SSO functionality.

Web settings.

The following are the updates that are required to the Custom Text settings for SSO.

From a Relius Administration workstation, open the Web Text Customization window (*Utilities | VRU/Web Administration | Web Customization | Text/Language*). In the “Custom” tab, scroll down to “**Common**” and expand it. Then click the “**SSOCONTROLOPTCD**” feature. Change this setting to “Y” to allow the SSO to be used in a manner where a web UserID and password is not required in the Relius database. **This option allows SSO to be setup so that the client application is the only entry point into the Relius Web products.**



Optional Field Variables

The following is a list of optional variables that can be used. **Note that the names for all field variables are case-sensitive.**

name="SSNUM"	Self explanatory
name="PLANID"	Self explanatory
name="FIRSTPAGE"	use to open first page after SSO.
name="REQUESTID"	use with DVC to open existing payroll.
name="DVCSUBMITFINAL"	value excepted is URL use to take user back to this page when user has access to submit for final
name="DVCNOSUBMITFINAL"	same as above except user doesn't have access to submit for final
name="PAYENDDATE"	use to open existing DVC payroll
name="PAYENDDATE"	use to open existing DVC payroll
name="PAYFREQCD"	use to open existing DVC payroll.
name="SUTSTOKEN2"	if webcookie token was requested, use this instead.

TokenGenerator WSDL

Attached to this document is a WSDL file for the token generator.



TokenGenerator.wsdl