**INSTRUCTIONS TO SETUP SAML ASP.NET CONNECTOR**

Contents

[Introduction 1](#_Toc482377055)

[Advantages: 2](#_Toc482377056)

[Pre-requisites: 2](#_Toc482377057)

[A. Steps to deploy SAML ASP.Net Connector in IIS Manager. 2](#_Toc482377058)

[Step 1: To Add Application in IIS Manager 2](#_Toc482377059)

[Step 2: Steps to give authority to user to make changes in SAML Connector 3](#_Toc482377060)

[Step 3: To Start Application with HTTPS [not mandatory] 4](#_Toc482377061)

[B. Steps to configure the SAML ASP.Net Connector with your SP and IdP: 4](#_Toc482377062)

[Step 1: Configure your Identity Provider 4](#_Toc482377063)

[Step 2: Configure your Service Provider 5](#_Toc482377064)

[Step 3: Attribute Mapping 6](#_Toc482377065)

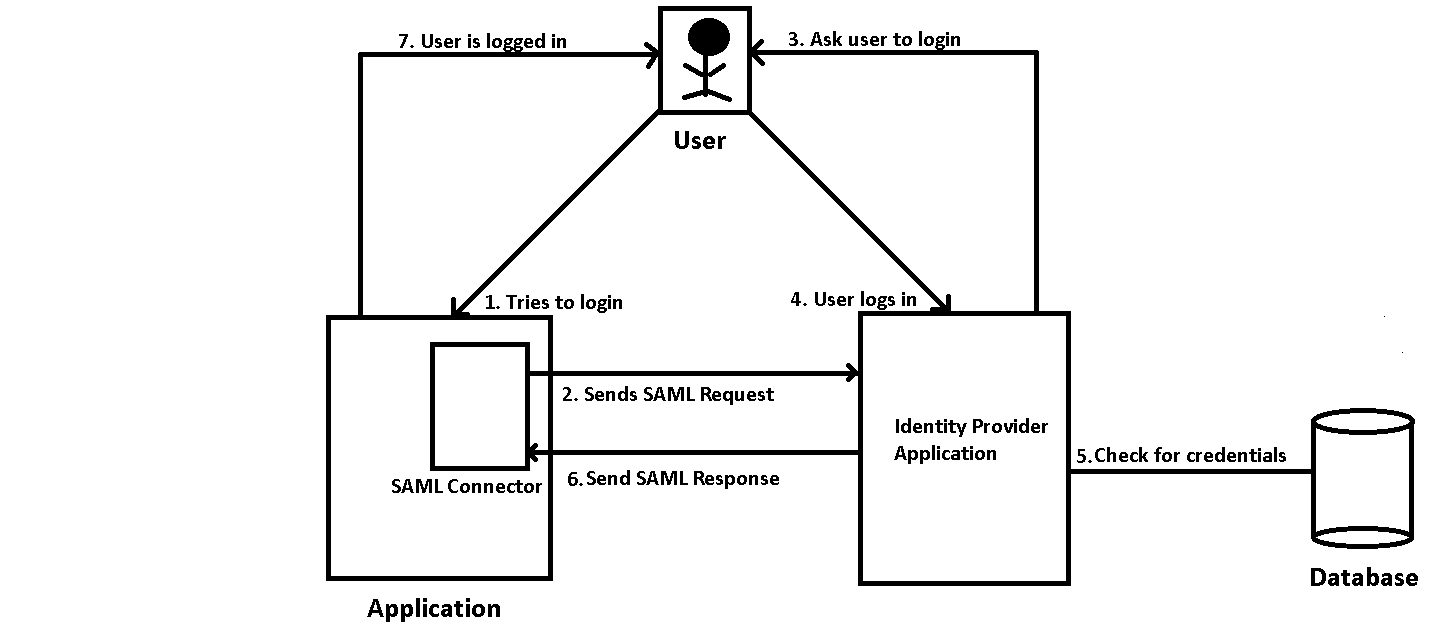
[Step 4: Login Setup 6](#_Toc482377066)

[Step 5: SSO Sign In Settings 7](#_Toc482377067)

[C. Single Sign On to your App: 8](#_Toc482377068)

# Introduction

SAML ASP.Net Connector is the way to go to make your application act as a full-fledged SAML 2.0 Service Provider.  
It provides Single Sign-On Service between your IdP and the ASP.NET application.



SAML completely eliminates the need for passwords and instead uses standard cryptography and digital signatures to pass a secure sign-in token from an identity provider such as: **miniOrange, ADFS, onelogin**, **Google Apps etc.** to ASP.Net application.

# Advantages:

* Secure your application
* Reduce development costs
* Reach greater and faster adoption.

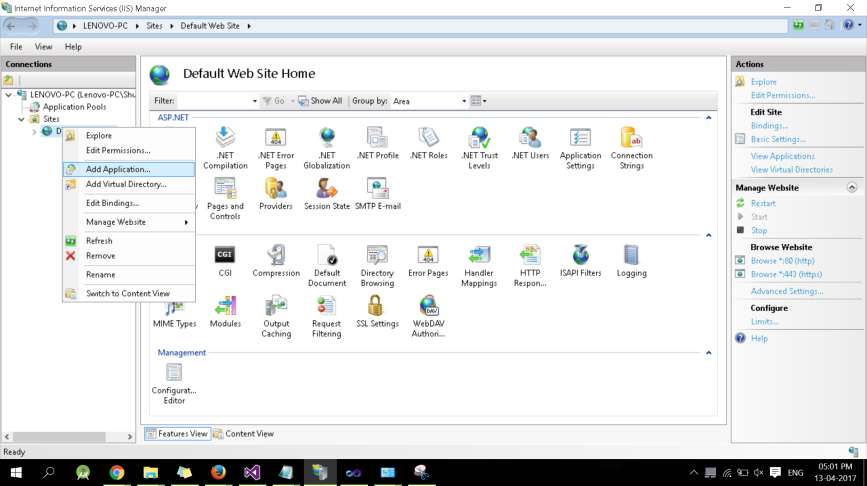
# Pre-requisites:

* To work with ASP.Net Connector you need ASP.Net Application.
* While deploying you SAML ASP.Net Connector, remember that other than **IIS Manager** no other server should be running for e.g. Apache, tomcat etc.
* Before configuring your SP and IdP in the saml asp.net connector, follow [**B. Step 5**](#_Step_5:_SSO)in your **Login Application.**
* Same way deploy your ASP.Net application as you deploy connector.
* All code related changes would be done before copying the application and connector in the IIS Manager.

# Steps to deploy SAML ASP.Net Connector in IIS Manager.

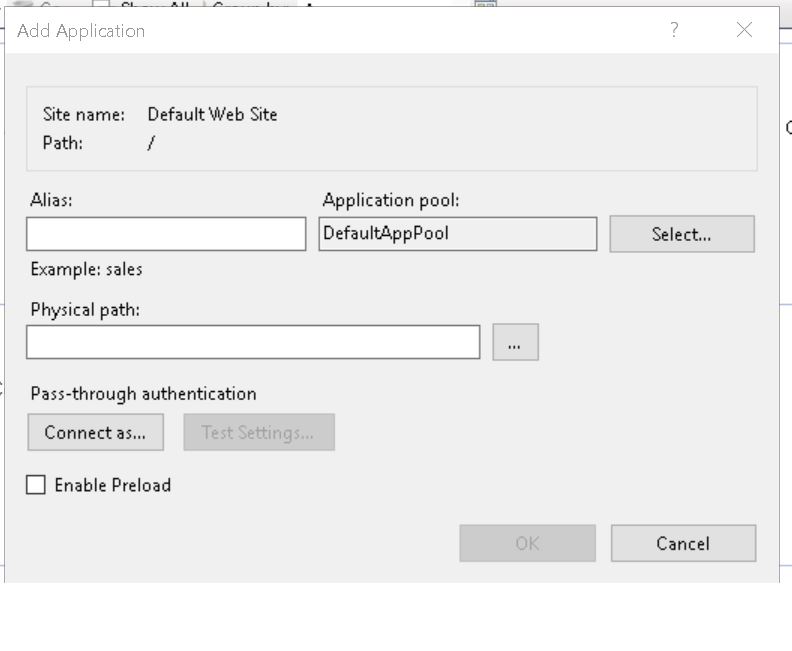
## Step 1: To Add Application in IIS Manager

* Extract **saml\_asp.net\_connector.zip** and Copy **saml\_asp.net\_connector** folder to path- **C:\inetpub\wwwroot.**
* Open IIS manager
* In left panel right click **Default Web Site** and click **Add Application**.



* As given below provide Alias Name that is a webpage Address Name e.g. ***Alias Name***
* Give **Physical path** where you have copied the Application i.e.

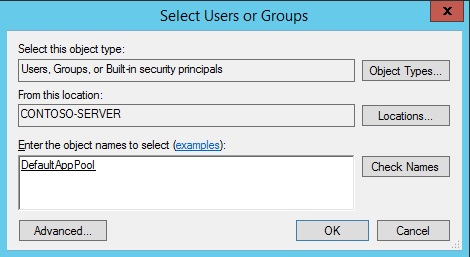
*C:\inetpub\wwwroot\**saml\_asp.net\_connector.*



## Step 2: Steps to give authority to user to make changes in SAML Connector

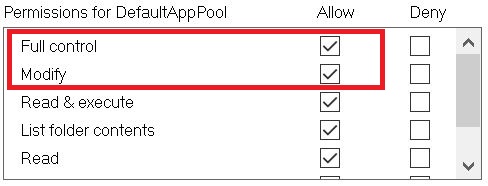
**(web.config File)**.

* Go to this path *C:\inetpub\wwwroot\**saml\_asp.net\_connector*
* Right click the file and select **Properties**
* Select the **Security** tab
* Click the **Edit** button and then **Add** button
* Click the **Locations** button and make sure that you select your computer.

[](https://media-www-iis.azureedge.net/media/7427483/application-pool-identities-selectusers2.jpg)

Enter **IIS AppPool\DefaultAppPool** in the **Enter the object names to select** text box.

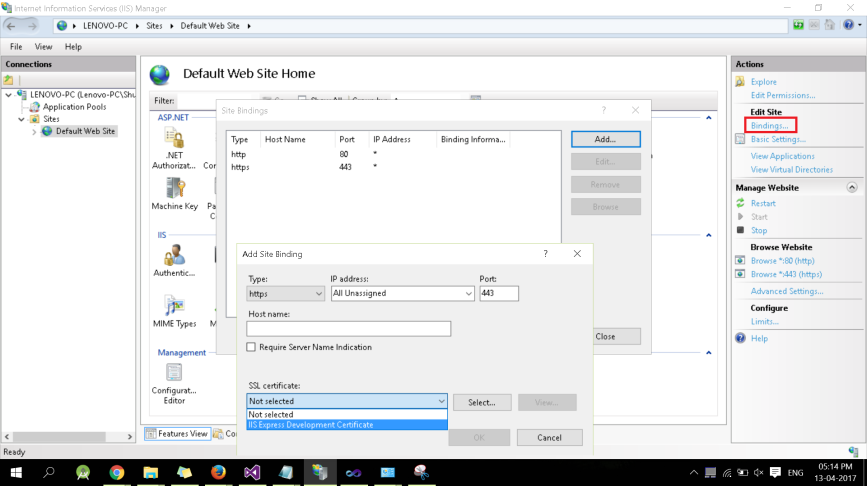
* Click the **Check Names** button and click **OK**.
* After following the steps given above, tick the box in the **Permission for DefaultAppPool** container as shown in the figure below:



* Click **Ok** button

## Step 3: To Start Application with HTTPS [not mandatory]

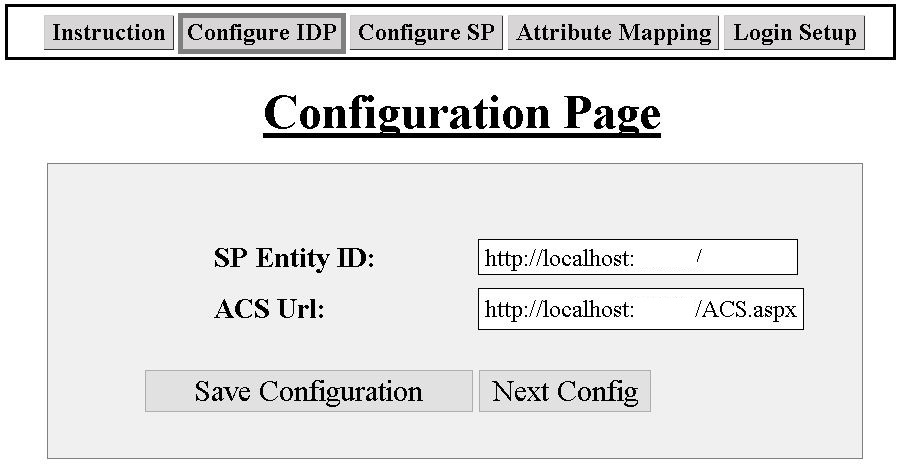
* To make Application start with https select **Default web site** and at right side click **Binding**
* Click on **Add** change **http** to **https**



# Steps to configure the SAML ASP.Net Connector with your SP and IdP:

## Step 1: Configure your Identity Provider

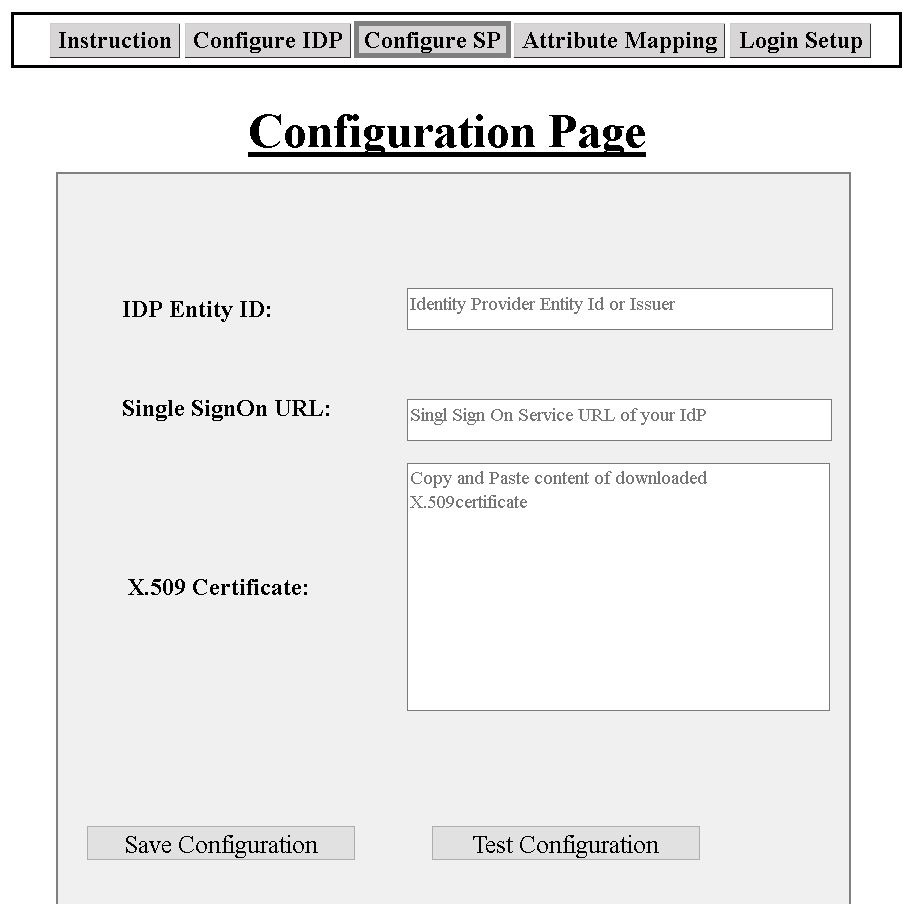
* Copy this generated **SP Entity ID** to your IdP settings. Also, you can edit this and add your own custom **Entity ID**.
* Copy this **ACS URL** to your IdP settings e.g. miniOrange, LDAP, Google APP, onelogin, etc.
* Click **Save Configuration** button to save your configuration.
* After saving the SP details in **Configure IDP** click on **Next Config** to configure SP.



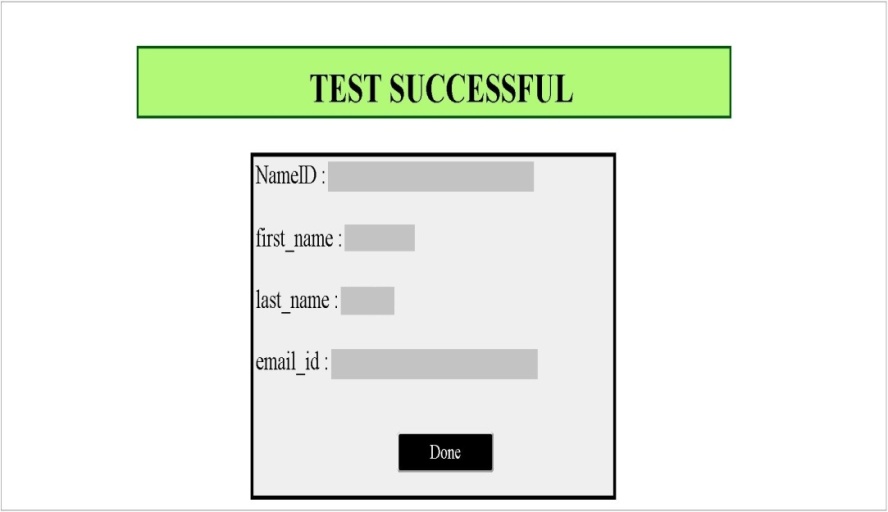
## Step 2: Configure your Service Provider

Provide the following details:

* Using IdP information user could add details in **Configure SP**.
* Provide the required settings (i.e. **IdP Entity ID, IdP Single SignOn Url, X.509 certificate)** in the Connector.
* Click **Save Configuration** to save your IdP details.
* Then, click **Test Configuration** button.

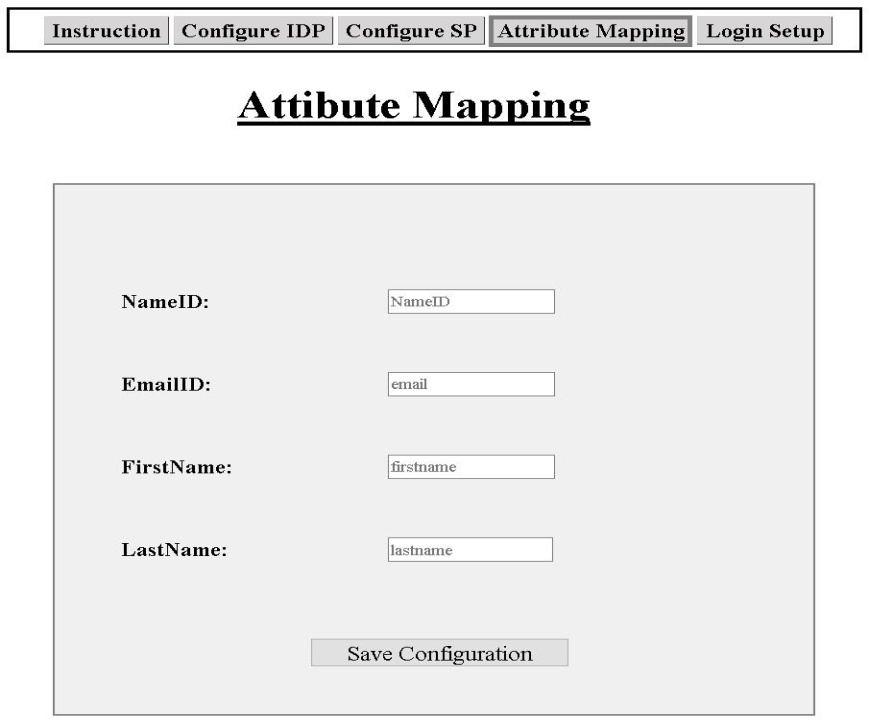


* This screenshot shows the attributes that are received and are mapped by attribute mapping (i.e. NameID, email, firstname, lastname).



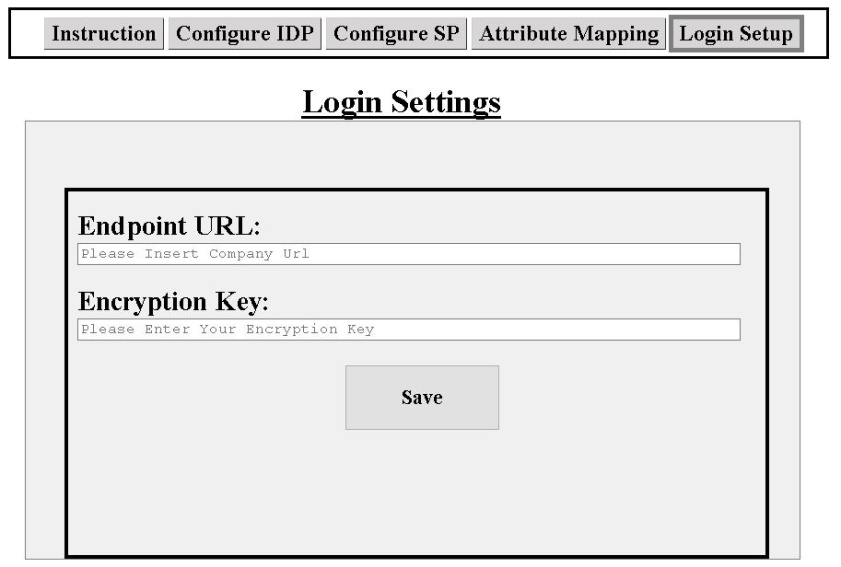
## Step 3: Attribute Mapping

* Attribute Mapping helps you to get user attributes from your IdP and map them to your user attributes in SP.
* In Attribute Mapping details like **NameID** and **Email** as shown in Fig. **Test Successful** are mapped to **NameID** and **Email** respectively i.e. **Attribute: NameID, Email, FirstName, LastName**.



## Step 4: Login Setup

* Provide your **Endpoint URL** in **Login Setup** where you will read your SAML response.
* Add 16-character **Encryption Key** which will be used to encrypt response and **same key** will be used to **Decrypt** the response in login ASP.Net application.
* Click on **Save** button.



## Step 5: SSO Sign In Settings

* Add below code in **web.config** file of your **asp.net application**:

<appSettings>

<add key="Key Name" value="https://localhost/[Connector Alias Name]/request.aspx" />

</appSettings>

* Add the SAML login link by adding code to your **Login UI** part:

<a href=”<%=ConfigurationManager.AppSettings["key Name"] %>”>Login with IdP</a>

* Also, add below code to **Decrypt** the attributes at your login side(**aspx.cs file**):

String username = Decrypt(Request.Form["Name"].Trim());

FormsAuthentication.RedirectFromLoginPage(username,true);

* Add same 16-character key which is used in Encryption code
* Decrypt Code**(aspx.cs file)**

private string Decrypt(string ciphertext) {   
  
        string EncryptionKey = “ **Encryption Key**”  
        byte[] cipherBytes = Convert.FromBase64String(ciphertext); using (Aes encryptor = Aes.Create())  
{   
          Rfc2898DeriveBytes pdb = new Rfc2898DeriveBytes(EncryptionKey, new byte[]  
         { 0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d, 0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76 });   
          encryptor.Key = pdb.GetBytes(32);   
          encryptor.IV = pdb.GetBytes(16);   
       using (MemoryStream ms = new MemoryStream())  
         {   
            using (CryptoStream cs = new CryptoStream  
            (ms, encryptor.CreateDecryptor(), CryptoStreamMode.Write))  
              {   
                  cs.Write(cipherBytes, 0, cipherBytes.Length);   
                  cs.Close();   
               }   
                  ciphertext= Encoding.UTF8.GetString(ms.ToArray());  
            }  
        }  
            return ciphertext;      
}

# Single Sign On to your App:

* After completing above steps, you will find the **link on your ASP.Net application**.
* After clicking the login link, it will redirect you to IdP login page where you have to enter your login credentials.
* After entering the credentials into your IdP, you would be logged into your ASP.Net application.