Leveraging ADFS in your .NET Web application

Introduction

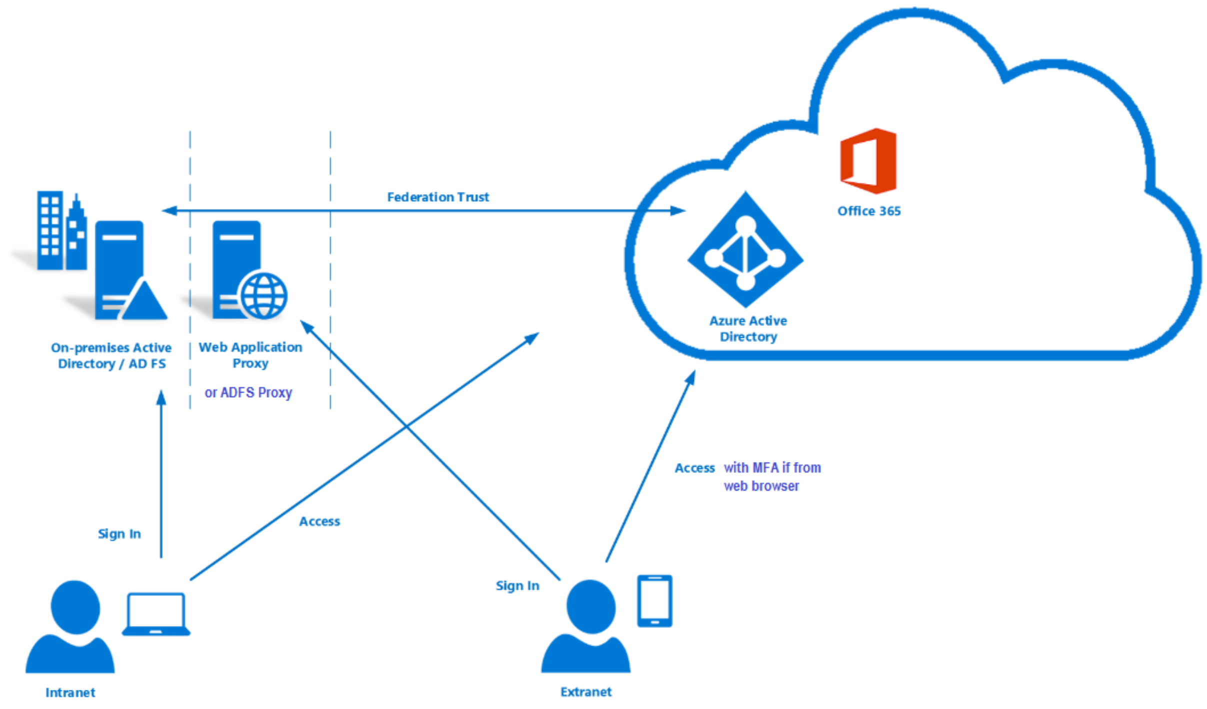
In this series of blog posts, we will walk through scenarios where you can leverage ADFS in your .NET web applications for authentication. In the first blog, we will walk through enabling ADFS authentication in a classic ASP.NET web application. In future posts, we will build on this knowledge gained to enable ADFS authentication in an Sitecore enterprise grade application.

I will encourage you to walk through and create the sample project as we go along. If you prefer to clone the final code, the link is available at the end of the blog.

What is ADFS?

Active Directory Federation Services (ADFS) is a feature and web service in the Windows Server Operating System that allows sharing of identity information outside a company's network. It authenticates users with their usernames and passwords.

Users can access some applications (for example Office 365, Salesforce.com, other custom web apps, etc.) without being prompted to provide login credentials again. These applications can be on premise, on the cloud, or even hosted by third parties. Below is a diagram to explain the concept. Irrespective of where these applications live or who they belong to, the user accounts can be maintained by the administrator from a single place, namely Active Directory (AD)



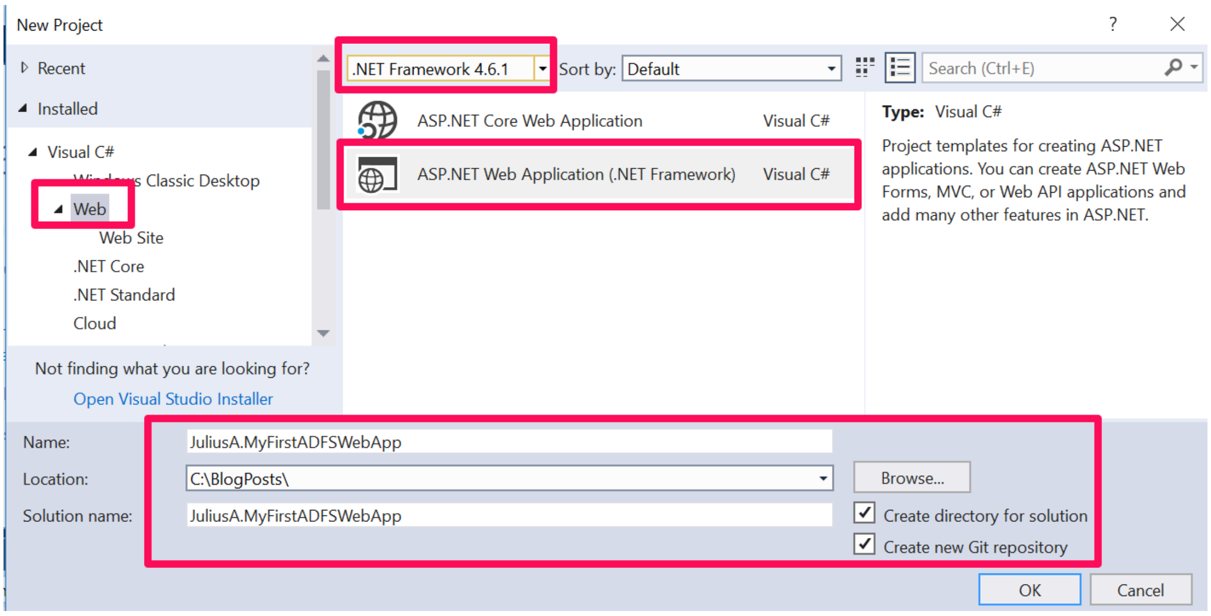
ADFS architecture 1

As the above architecture diagram depicts, identity information is being shared between trusted partners beyond the boundary of an organization, hence the name Federation.

Create a test .NET web application with C#

In the following section, we will walk through steps to create a test application using Visual Studio 2107 Professional Edition (VS2017). Please note you can use any other version of Visual Studio with .NET Framework 4.5 and above, including the free Community Edition, available from https://www.visualstudio.com/downloads/

Launch VS2017 and select to create a new project. In the dialog like one below, ensure you select a Web category from the Installed templates. I have left the default .NET Framework 4.6.1 and selected the Asp.NET Web Application (.NET Framework) project template. Finally, I have specified a location on my local development machine where to save my project. You will also notice I have opted to create a new Git repository for my project. Then click OK.



New Project

.NET Framework 4.6.1 — Sort by: Default — Search (Ctrl+E)

- Recent
- Installed
  - Visual C#
    - Windows Classic Desktop
    - Web
      - Web Site
    - .NET Core
    - .NET Standard
    - Cloud

Not finding what you are looking for?
Open Visual Studio Installer

ASP.NET Core Web Application	Visual C#
ASP.NET Web Application (.NET Framework)	Visual C#

Type: Visual C#

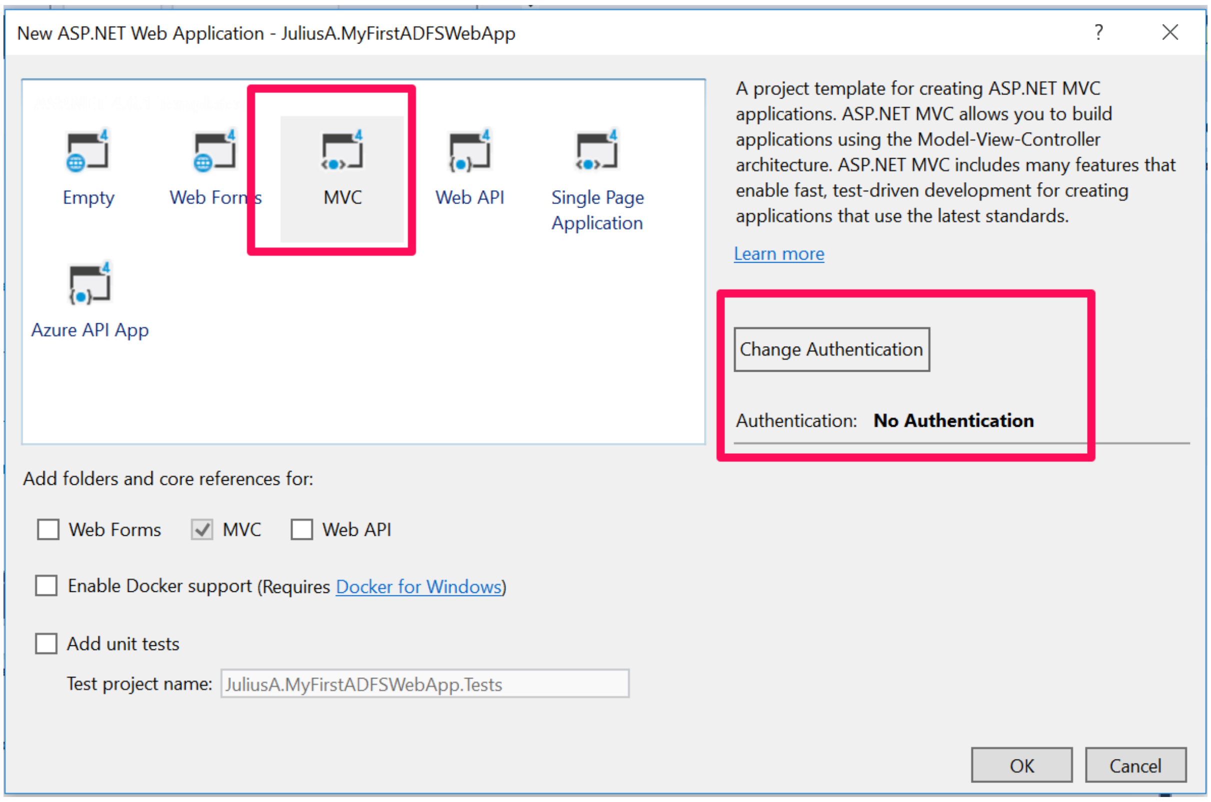
Project templates for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.

Name: JuliusA.MyFirstADFSWebApp
Location: C:\BlogPosts\
Solution name: JuliusA.MyFirstADFSWebApp

☑ Create directory for solution
☑ Create new Git repository

Screenshot 1 – Create a new project in VS2017

This will launch another dialog box like the one below where you will select the MVC project template, then click on Change Authentication button, to configure the Authentication settings for our test app.



New ASP.NET Web Application - JuliusA.MyFirstADFSWebApp

A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.

Learn more

Change Authentication

Authentication: **No Authentication**

Add folders and core references for:

☐ Web Forms ☑ MVC ☐ Web API

☐ Enable Docker support (Requires Docker for Windows)

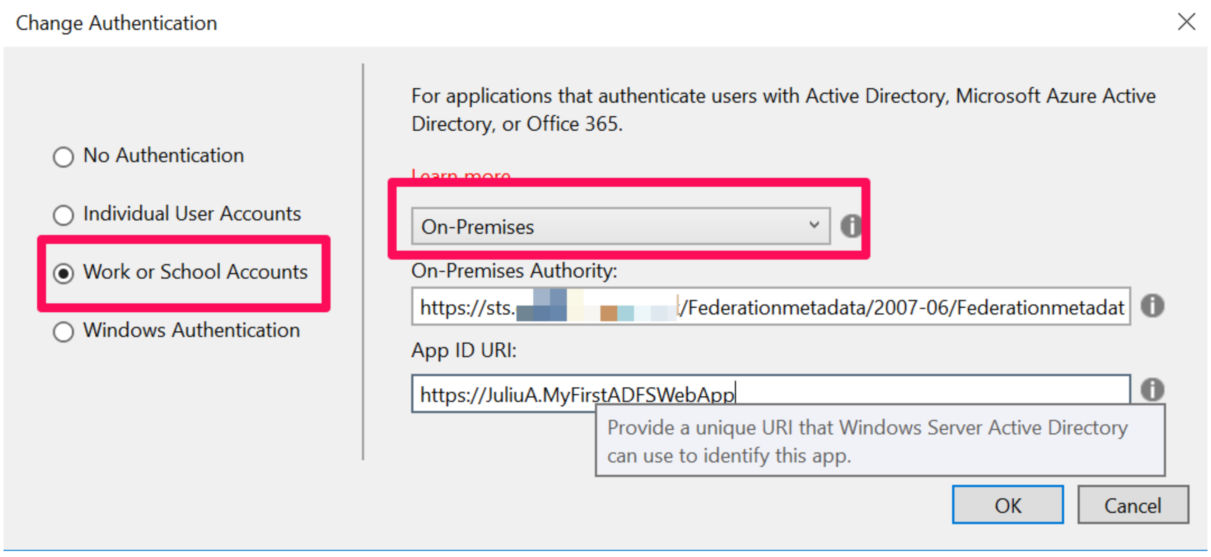
☐ Add unit tests

Test project name: JuliusA.MyFirstADFSWebApp.Tests

Screenshot 2 - Project template and Authentication

Clicking Change Authentication will launch the Change Authentication dialog like the one shown below. On this dialog:

- select Work or School Accounts option,
- select the On-Premises item from the drop-down list. This option will require you to specify the ADFS on premises authority and your application ID URI. You will need to ask your IT services for the public URL of your FederationMetadata.xml on the identity server. This usually in the form of: https://sts.<YOUR_ADFS_PUBLIC_DOMAIN>/Federationmetadata/2007-06/Federationmetadata.xml
- For the App ID URI, you must enter an identifier for your app. This is not a real URL address, just a unique identifier, for example https://JuliuA.MyFirstADFSWebApp


Change Authentication

- No Authentication
- Individual User Accounts
- [✓ Work or School Accounts]
- Windows Authentication

For applications that authenticate users with Active Directory, Microsoft Azure Active Directory, or Office 365.

Learn more

On-Premises

On-Premises Authority:

https://sts.[redacted]/Federationmetadata/2007-06/Federationmetadat

App ID URI:

https://JuliuA.MyFirstADFSWebApp

Provide a unique URI that Windows Server Active Directory can use to identify this app.

Screenshot 3 - Change Authentication

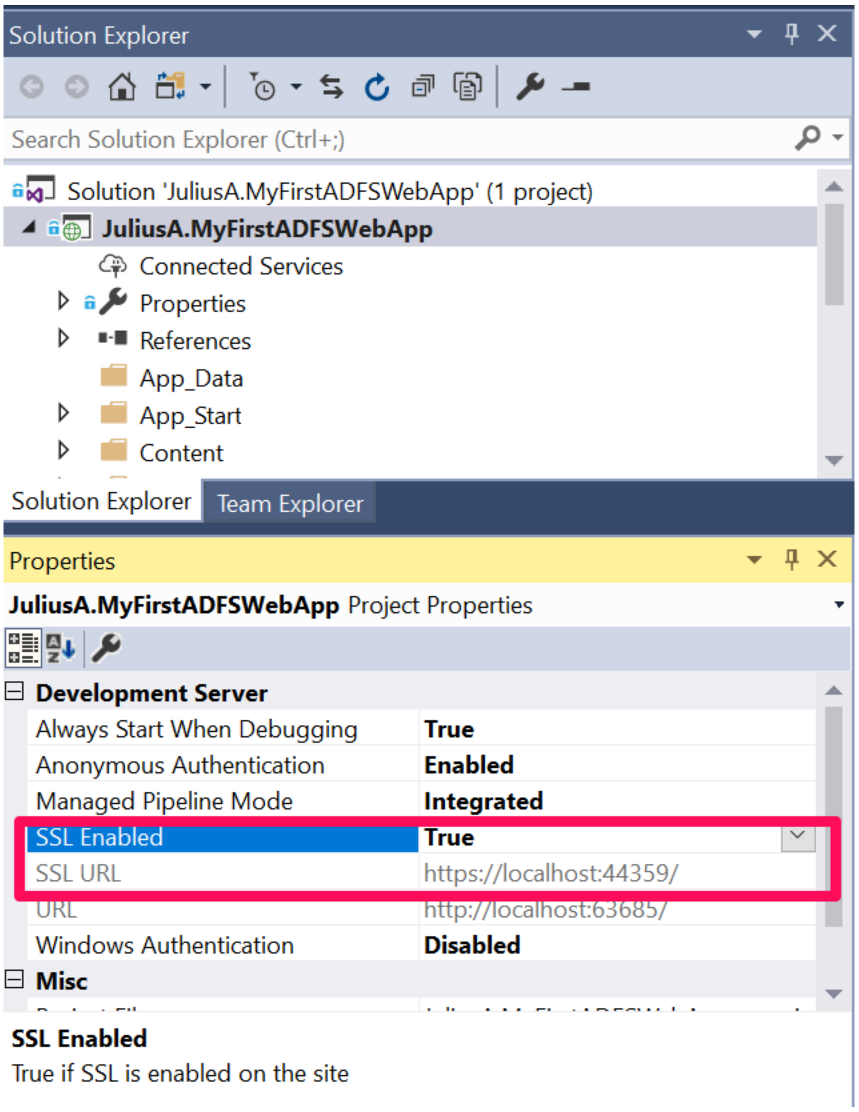
Important: The App ID URI identifies your app with the On-premises ADFS identity server. This same App ID must be registered on the ADFS identity server by your IT services as a **Relying Party Trust** identifier (sometimes known as **Realm**), so that the server will accept requests.

Finally, click OK on the dialog to accept your changes.

Then proceed to finish up the project creation process.

Edit project settings after creating project

You need to make sure your projects is running on HTTPs. To do this, update the project settings as shown below to enable SSL.



JuliusA.MyFirstADFSWebApp Project Properties

Development Server	
Always Start When Debugging	**True**
Anonymous Authentication	**Enabled**
Managed Pipeline Mode	**Integrated**
SSL Enabled	**True**
SSL URL	https://localhost:44359/
URL	http://localhost:63685/
Windows Authentication	**Disabled**
Misc	

SSL Enabled
True if SSL is enabled on the site

Screenshot 4 - Edit project settings to enable SSL

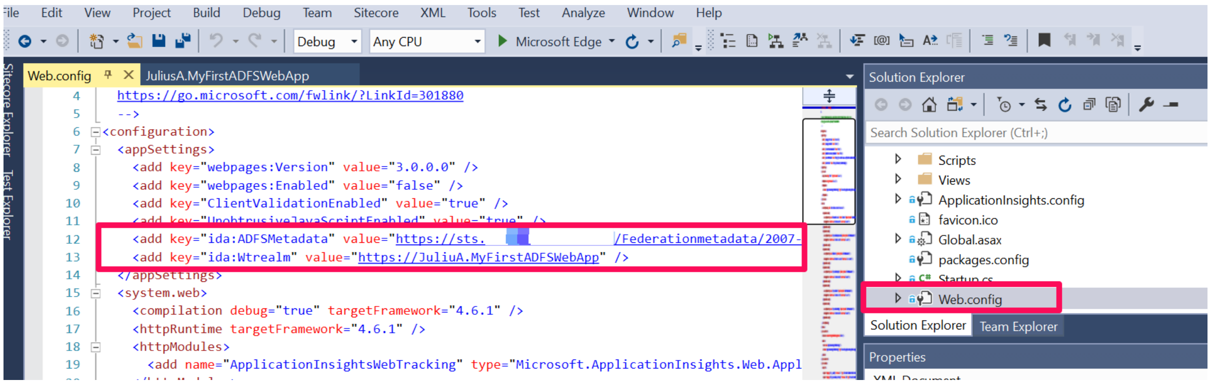
At this stage, try and compile and run your project. You can do this from Build -> Build Solution menu (or CTRL + SHIFT + B). Pay attention to your browser URL when it launches.

If you have done everything correctly, your web app should launch on https://localhost:44359/ (your local port may be different to mine), then be redirected to the ADFS URL you configured. Then you will end up on the ADFS login page configured by your IT services. The credentials you enter on this login page will be validated on the ADFS. If successfully authenticated, will then ADFS will redirect you back to your test web app. I will explain in more detail this process in the following sections.

Where is my ADFS Authentication code now?

If you are wondering where the authentication code resides, here are the details:

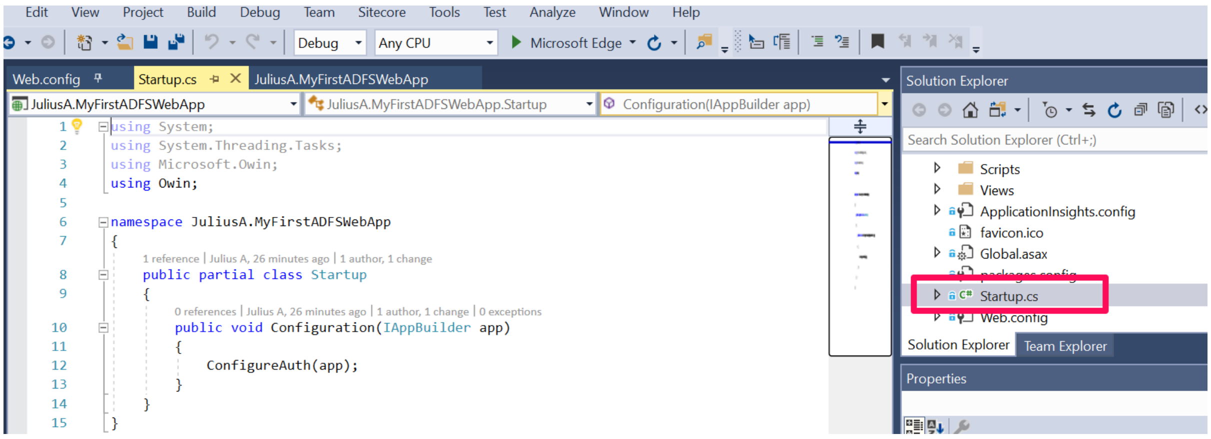
The App ID URI and the On-Premises Authority URL are stored in the <appSettings> section within the web.config file:



```xml
https://go.microsoft.com/fwlink/?LinkId=301880
-->
<configuration>
  <appSettings>
    <add key="webpages:Version" value="3.0.0.0" />
    <add key="webpages:Enabled" value="false" />
    <add key="ClientValidationEnabled" value="true" />
    <add key="UnobtrusiveJavaScriptEnabled" value="true" />
    <add key="ida:ADFSMetadata" value="https://sts.[redacted]/Federationmetadata/2007-
    <add key="ida:Wtrealm" value="https://JuliuA.MyFirstADFSWebApp" />
  </appSettings>
  <system.web>
    <compilation debug="true" targetFramework="4.6.1" />
    <httpRuntime targetFramework="4.6.1" />
    <httpModules>
      <add name="ApplicationInsightsWebTracking" type="Microsoft.ApplicationInsights.Web.Appl
```

Screenshot 5 - Web.config file showing ADFS app settings

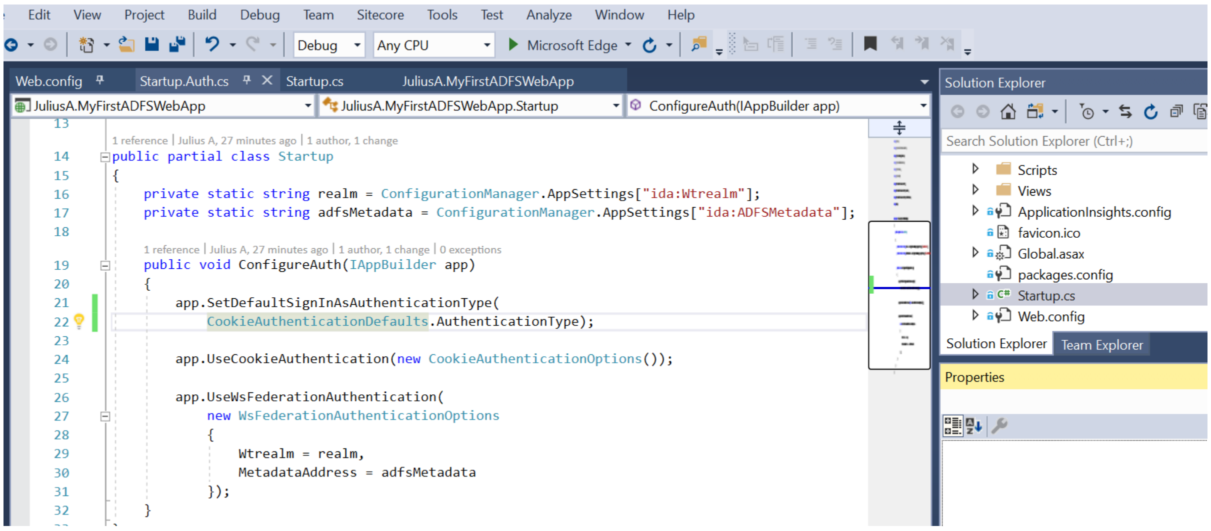
The code that challenges for the authentication is hosted within a Startup.cs class, at the root of the project. The ConfigureAuth method is defined in the partial class, as shown below



```csharp
using System;
using System.Threading.Tasks;
using Microsoft.Owin;
using Owin;

namespace JuliusA.MyFirstADFSWebApp
{
    1 reference | Julius A, 26 minutes ago | 1 author, 1 change
    public partial class Startup
    {
        0 references | Julius A, 26 minutes ago | 1 author, 1 change | 0 exceptions
        public void Configuration(IAppBuilder app)
        {
            ConfigureAuth(app);
        }
    }
}
```

Screenshot 6 - Startup.cs class



```csharp
    1 reference | Julius A, 27 minutes ago | 1 author, 1 change
    public partial class Startup
    {
        private static string realm = ConfigurationManager.AppSettings["ida:Wtrealm"];
        private static string adfsMetadata = ConfigurationManager.AppSettings["ida:ADFSMetadata"];

        1 reference | Julius A, 27 minutes ago | 1 author, 1 change | 0 exceptions
        public void ConfigureAuth(IAppBuilder app)
        {
            app.SetDefaultSignInAsAuthenticationType(
                CookieAuthenticationDefaults.AuthenticationType);

            app.UseCookieAuthentication(new CookieAuthenticationOptions());

            app.UseWsFederationAuthentication(
                new WsFederationAuthenticationOptions
                {
                    Wtrealm = realm,
                    MetadataAddress = adfsMetadata
                });
        }
```

Screenshot 7 - ConfigureAuth method

From the code snippets about, you can see the ConfigureAuth method configures your test application to use the ADFS settings previously saved in the web.config file.

The ADFS employs OWIN architecture. You can read more about this at http://owin.org/

Configure the On-Premises Identity Server (Job for your IT Services)

You will need to work with your IT services to ensure they have configured your test web app as a Relying Party Trust. For my case, I have provided my IT services my APP ID URI https://JuliuA.MyFirstADFSWebApp

Next Steps

Hope this post get you interested in leveraging ADFS authentication in your application. In my next blog post, I will build on this knowledge to apply ADFS authentication to a Sitecore application. Stay tuned for upcoming updates.

You can clone my sample web application from this location, https://github.com/JuliusAngwenyi/adfs

About the author

I am 2x Sitecore MVP, MSCD, and I have been developing with Microsoft .NET framework since its inception. I usually blog about everything .NET at https://360agileweb.wordpress.com/

This blog post was written on 28th May 2018