This document is deprecated

Latest user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

ADFS-Pro Authentication
User Guide

glanton
corporate web services

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

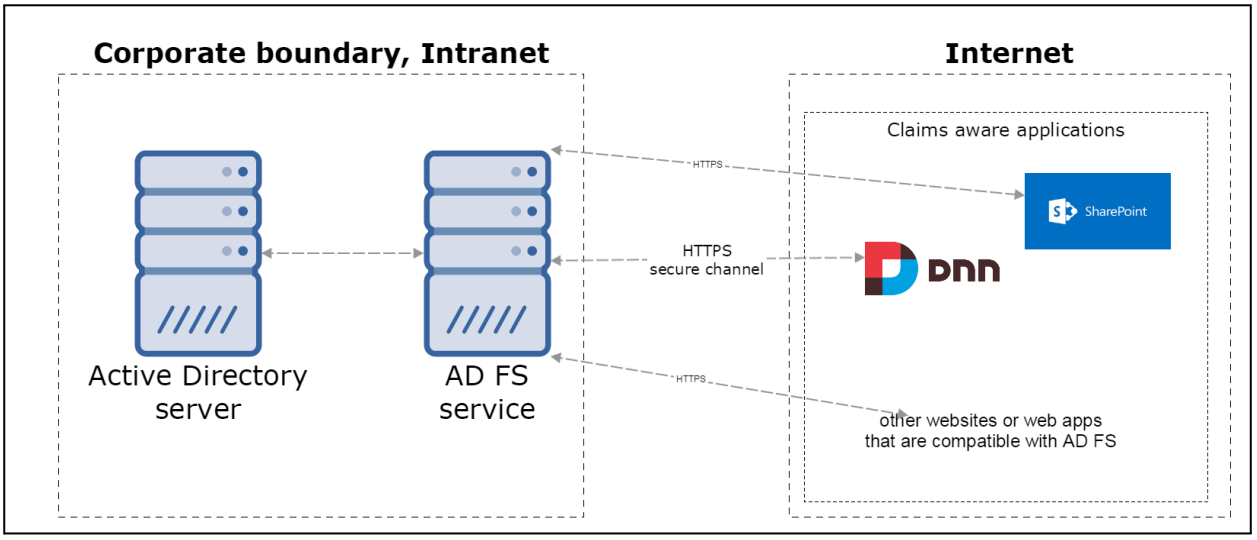
Introduction

Overview

Active Directory Federation Services (ADFS) is a component in Microsoft® Windows Server™ 2003 R2 (or higher versions) that provides authentication technologies. In details it allows authenticate user to a web application. ADFS is an official and mature tool, blessed by Microsoft.
To authenticate DNN user in AD FS a special DNN provider is required that will transform DNN into claims-aware application and makes possible to create federation between DNN and Active Directory. The DNN identities will rely on AD FS as an authorization backend. This document will show how to configure DNN that can take advantage of using AD FS.

Big picture

AD FS is an **identity mechanism** that allows access for people that are outside of the corporate boundary. In the secure way Active Directory resources (like identities) are exposed for web apps, that are hosted somewhere in the Internet.
One of the possible scenarios is described below. There is an on premise Active Directory placed in the corporate Intranet, and the web apps hosted outside of the corporate. Web apps are on the Internet whereby their access is opened for all. But in this case if someone wants to sign in to that app, his credentials are validated against the AD user store. This validation happens in a secure manner.



Target audience

Solution described in this document is targeted to:

- Active Directory admins who want's quickly add DNN website to their existing web app ecosystem.
- Companies that want to have good CMS website for their employees.

Use case - Company Blog

Blogs are valuable marketing tool for companies. Blogs can educate customers, build trust, and even bring in new leads.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

Company overview

Let say that we have big company. Company has employees whose identities are located in the Active Directory. It's an on-premise Active Directory system and access from the Internet is protected by the firewall. Security is very important for that company.
Company want's to have a blog. Blog will be for employees, customers and potential clients. Corporate admin doesn't want to create new accounts for users who needs add blog posts or blog comments. On the other hand employees doesn't want to have another username and password just for using company blog.

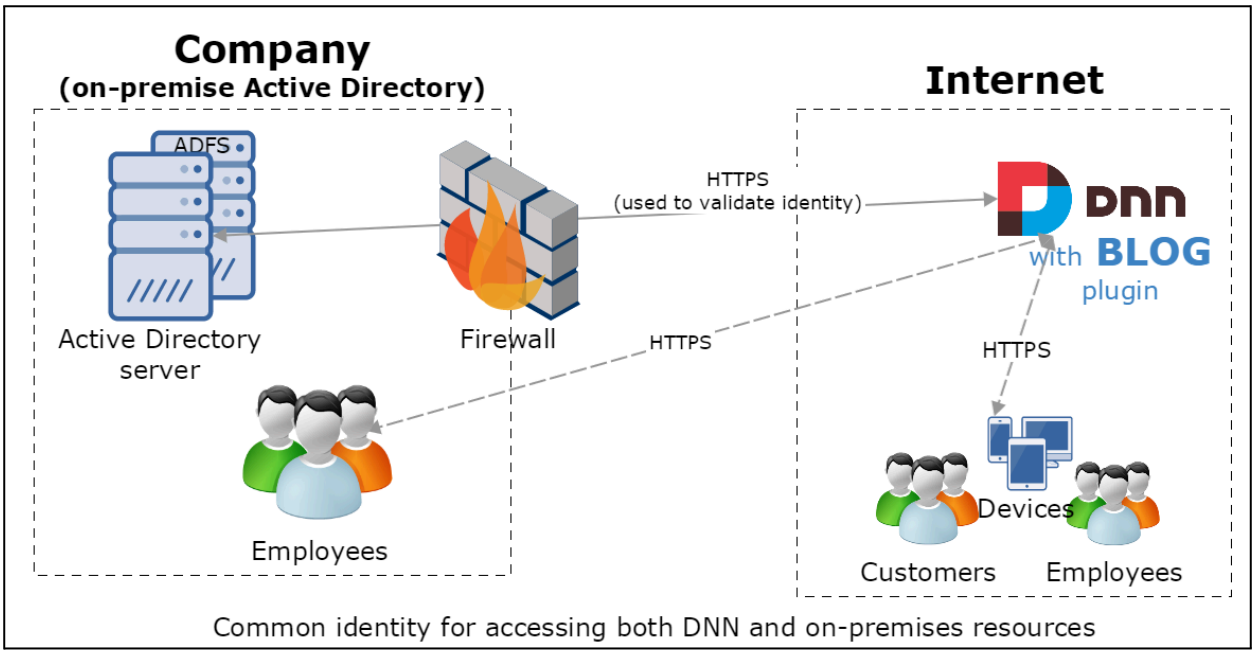
Requirements

Solution must meet following requirements:

- cost-effective solution,
- easy to maintain,
- accessible from corporate Intranet and from Internet (outside of the office),
- accessible from mobile devices (mobile friendly),

Solution

To achieve all these goals we can create solution described on the figure below.



We have DNN that is hosted outside of the company. DNN has an blog plugin. All users employees/customers have access to that blog. Additionally corporate employees using their actual identities can sign-in to DNN and add content to the blog (posts or comments).
What is most important: corporate admin doesn't need to create any new accounts on the DNN for users that want's to add blogs, posts or comments.

Benefits

CMS integrated with employees

From a company perspective, in just a few steps you can install vanila CMS, where company users can sign-in using their current credentials. There is no need to create new username/password for employees.

Outsource authentication

"ADFS-Pro Authentication" give you ability to outsource authentication process from DNN to the Active Directory. Authentication can be outsourced to any other security token service (STS) that is using the WS-Federation protocol like: Microsoft Azure Access Control Service (ACS), Identity Server, IBM Tivoli, Thinktecture, etc.

Multifactor authentication

ADFS can be configured to use with external authentication providers. This gives you ability to add second authentication factor, for example security code in mobile message. This will dramatically improves DNN security. For more information about additional authentication methods click here.

New authentication mechanisms

IT administrator can choose what authentication methods are used for DNN, based on the network location from which they access protected resources. For example administrator can mandate the use of more secure authentication methods for access requests from the extranet.
They can also enable device authentication for seamless second-factor authentication. This ties the user's identity to the registered device that is used to access the resource, thus offering more secure compound identity verification before protected resources are accessed.

One set of credentials

Corporate employees use a single set of credentials across all applications that they are using. One credential set to access: DNN, Salesforce, Office 365, etc.

Take away responsibility from DNN

DNN application that is using AD FS, is no longer responsible for the following:

- authenticating users, the authentication process is outsourced to external system like AD FS,
- storing user accounts and passwords, credentials are stored in Active Directory,
- integrating with other identity systems from other platforms or companies;

Additional Identity Providers

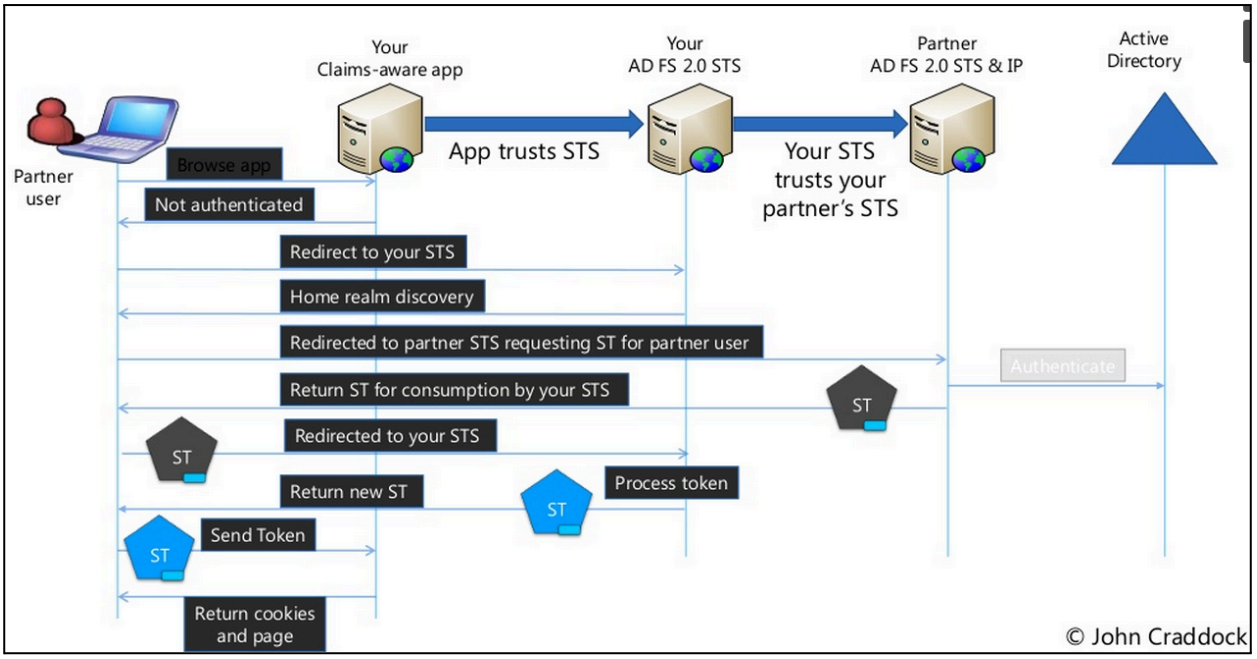
ADFS requires users to have an account in Active Directory or in one of the Identity Provider (IdP) that ADFS trusts. However, users may have no access to an Active Directory, but have accounts with other well-known IdP. These issuers typically are social networks and email providers. In this approach you can sign in to DNN using Facebook, Google or Windows Live account. For that scenarios an Microsoft Azure™ Access Control Service (ACS) must be implemented.

Prerequisites

To implement solution described in this document you need:

- DNN v7.3.4, or higher, that supports https protocol
- Modern web browser with enabled Java Script and cookies.
- Active Directory with installed AD FS service.
- DNN provider that consumes WS-Federation protocol, for example: 'ADFS-Pro Authentication'.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



AD FS configuration process

Overview

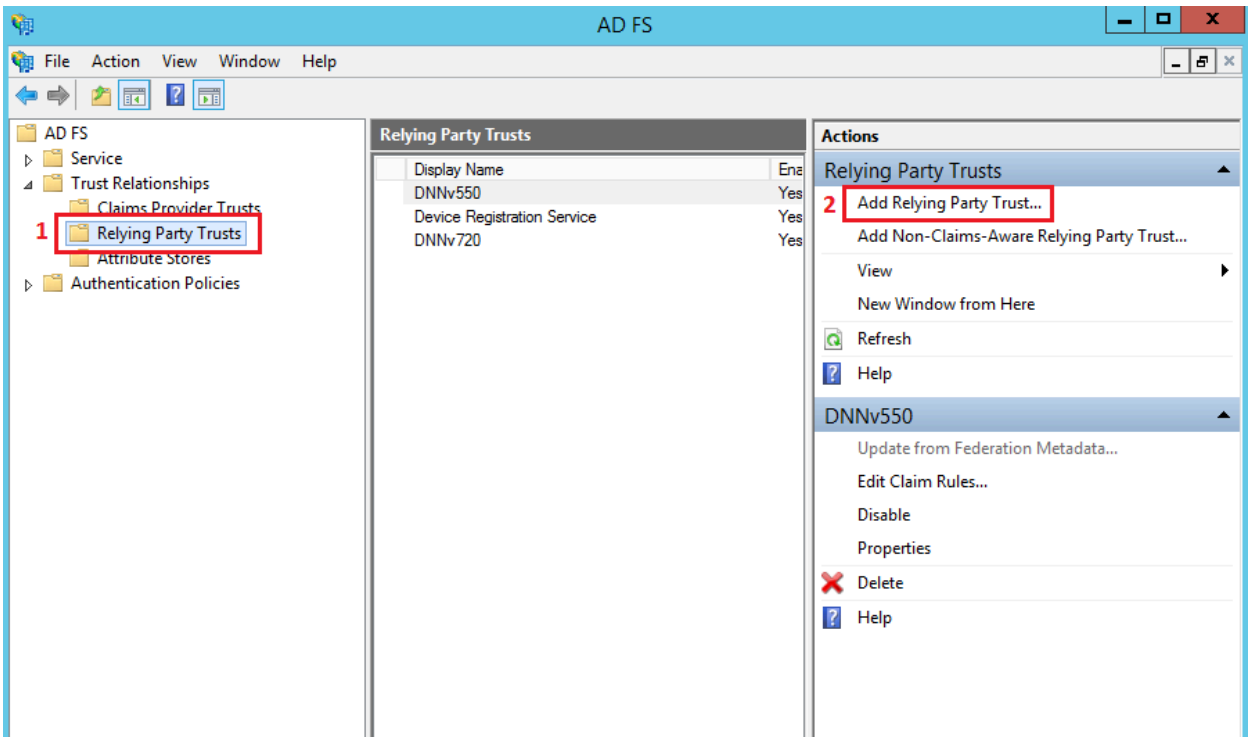
This chapter will describe all the steps necessary to configure Active Directory with DNN. First we describe how to create a "Relying Party". It's some kind of endpoint that will communicate with your DNN website. Then we show how to configure list of necessary claims. Claims will hold user profile fields that will be transported to DNN. If you don't have already installed AD FS service in your Active Directory system we refer to official Microsoft docs like here.

Configure Relying Party

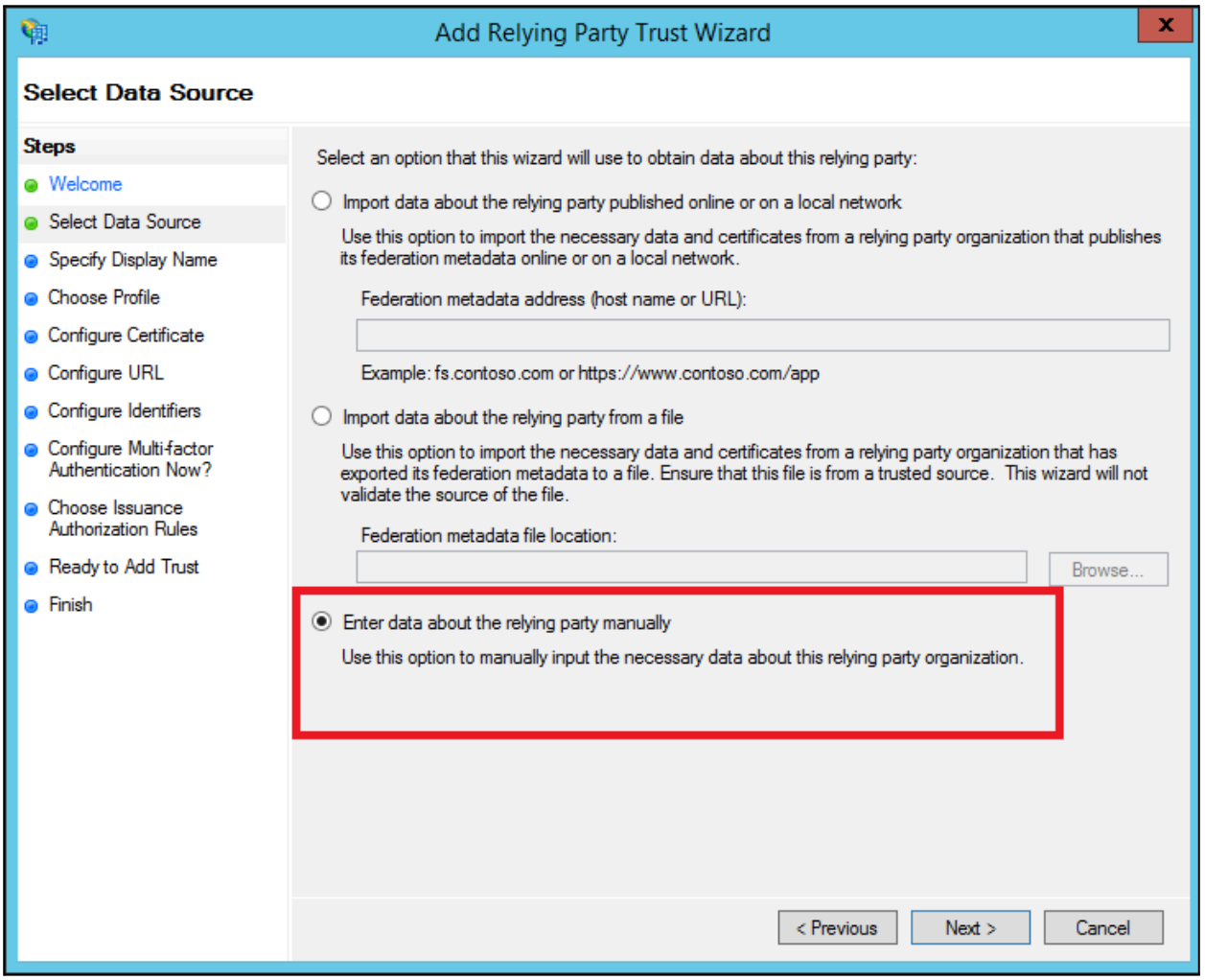
To set up connection between AD and DNN we need a Relying Party on the AD side. It's a entry point that will allow communication with your DNN application. Each DNN website requires separate "Relying Party". To add "Relying Party" execute following steps:

1. Open the AD FS Management console and select "Relying Party Trust", then select "Add Relying Party Trust…", see figure below.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

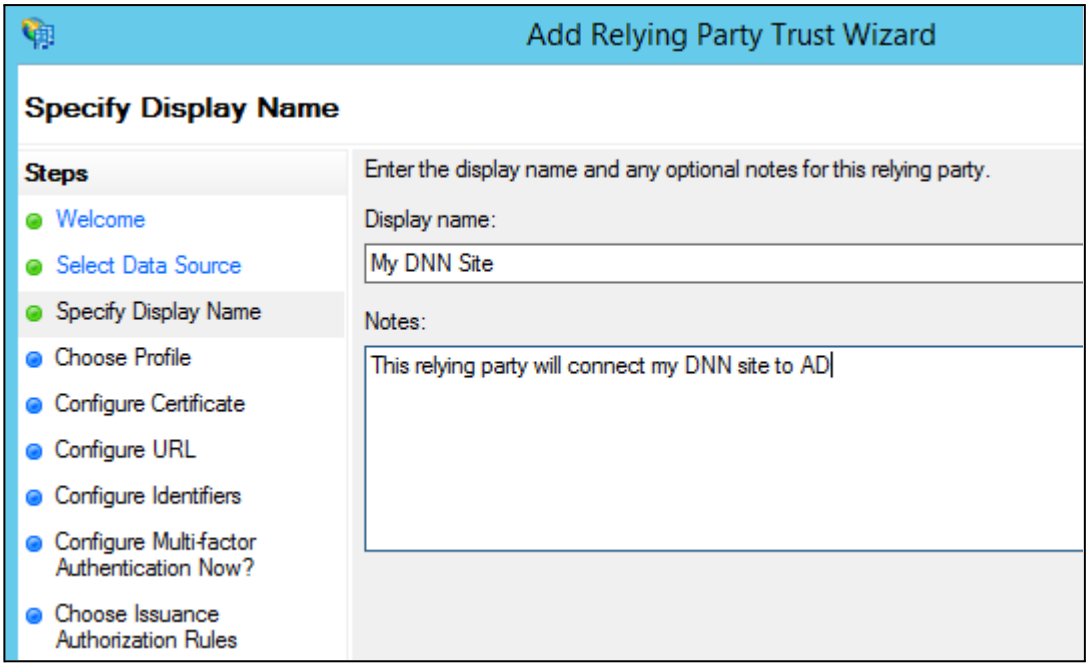


2. Select last option "Enter data manually.." to manually configure new Relying Party Trust, see figure below.

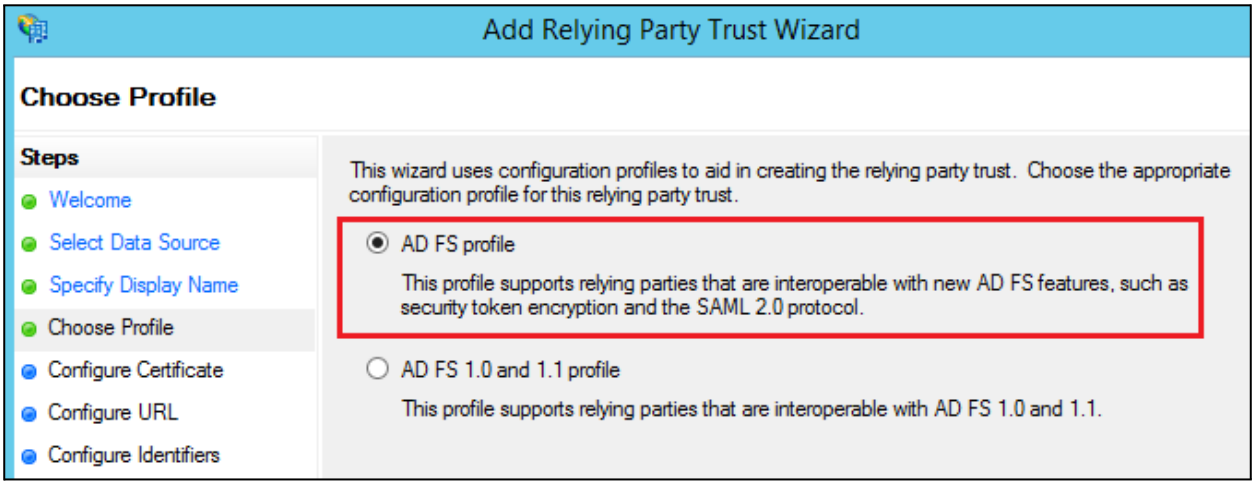


This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

3. Specify display name.

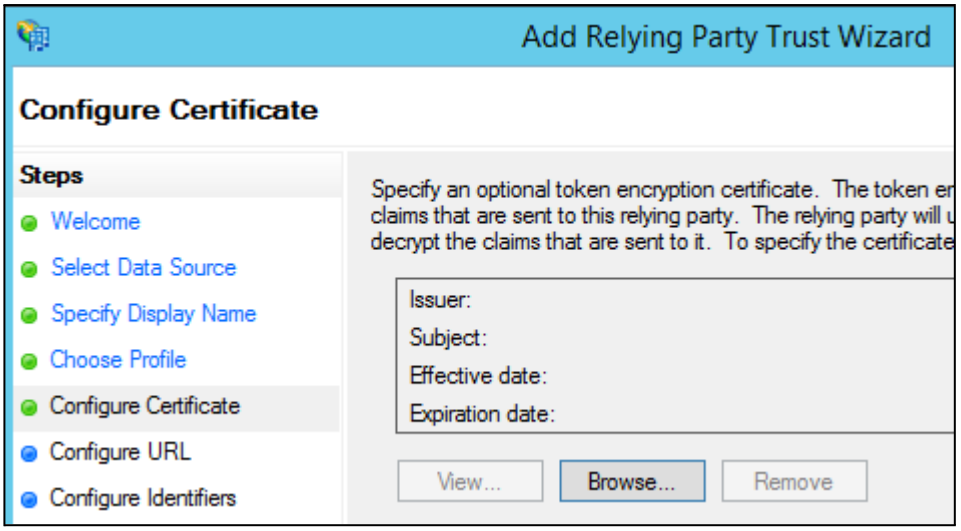


4. Select AD FS profile.

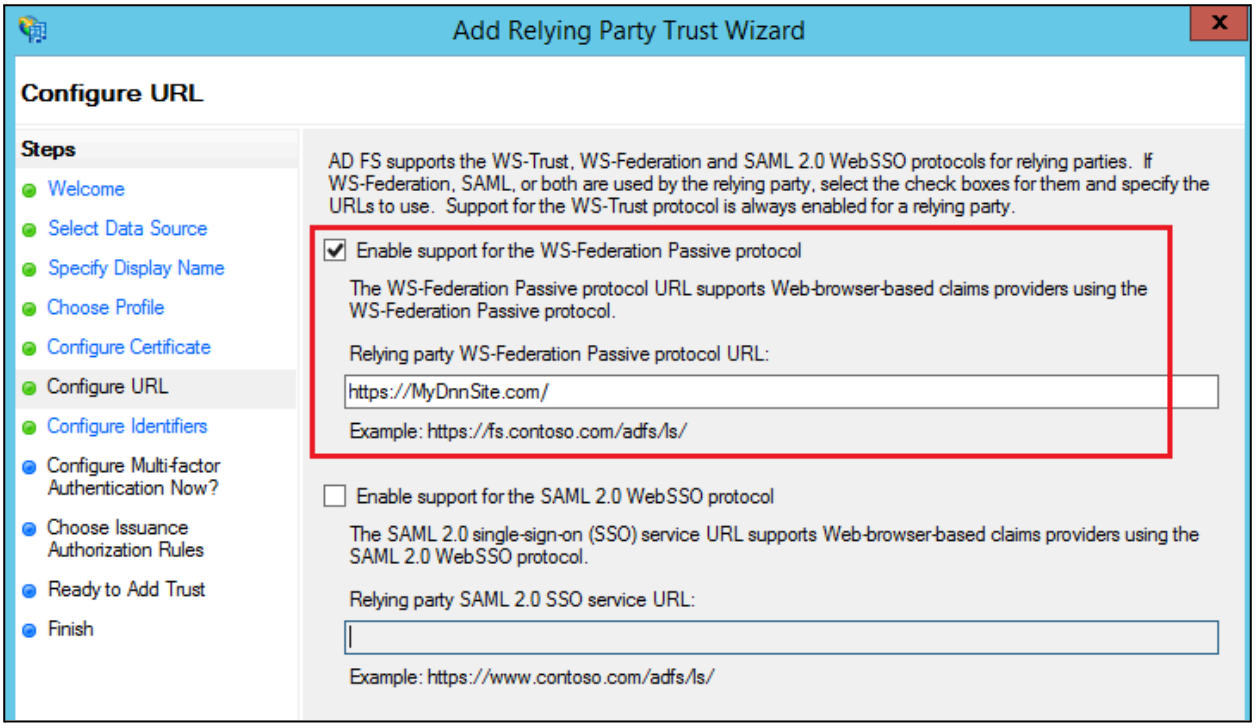


5. Do not configure additional certificate for now, just click next.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

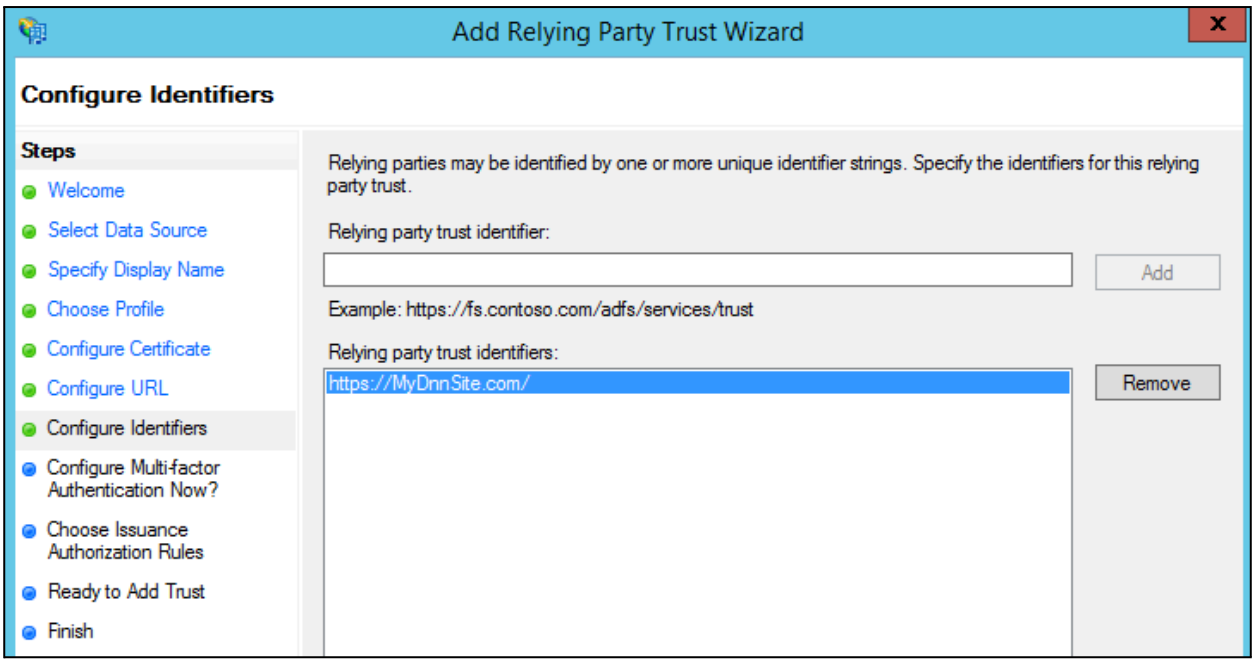


6. Enter WS-Federation endpoint address. It's your DNN website url, usually with the "/" at the end.

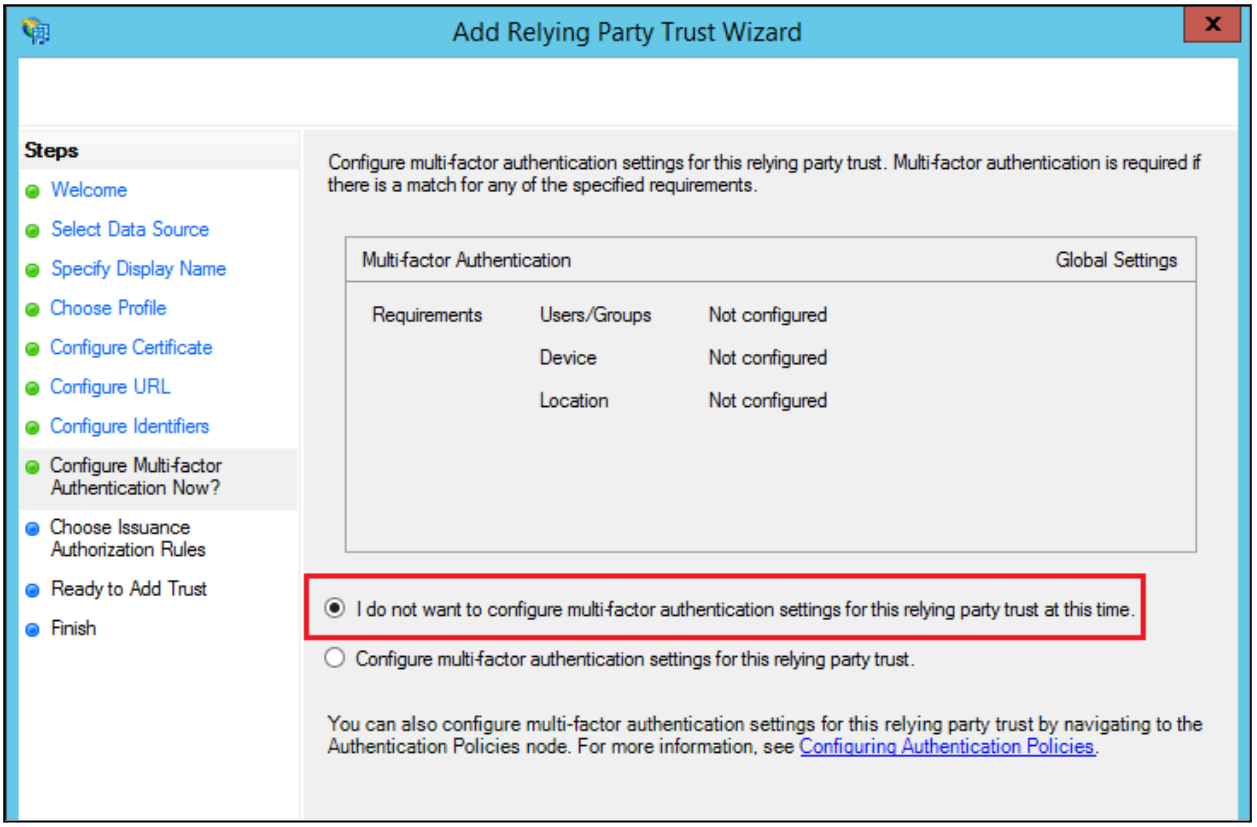


7. Add Relying Party identifier. It's your DNN website url, usually with the "/" at the end.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

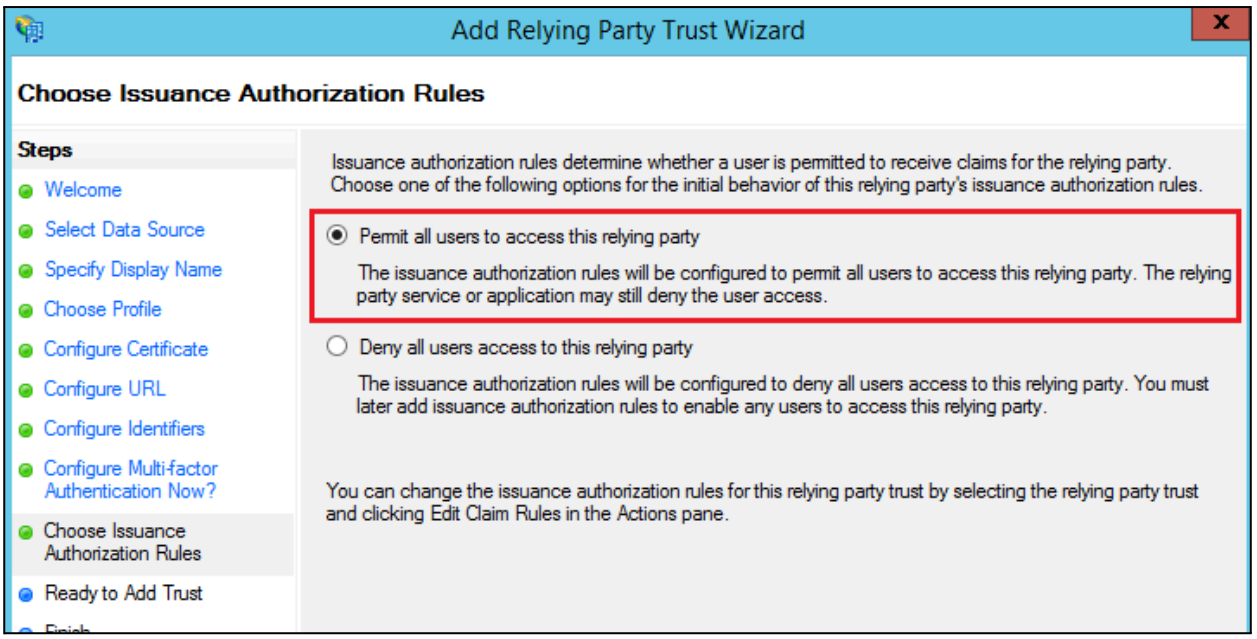


8. Do not configure Multi-factor authentication for now.

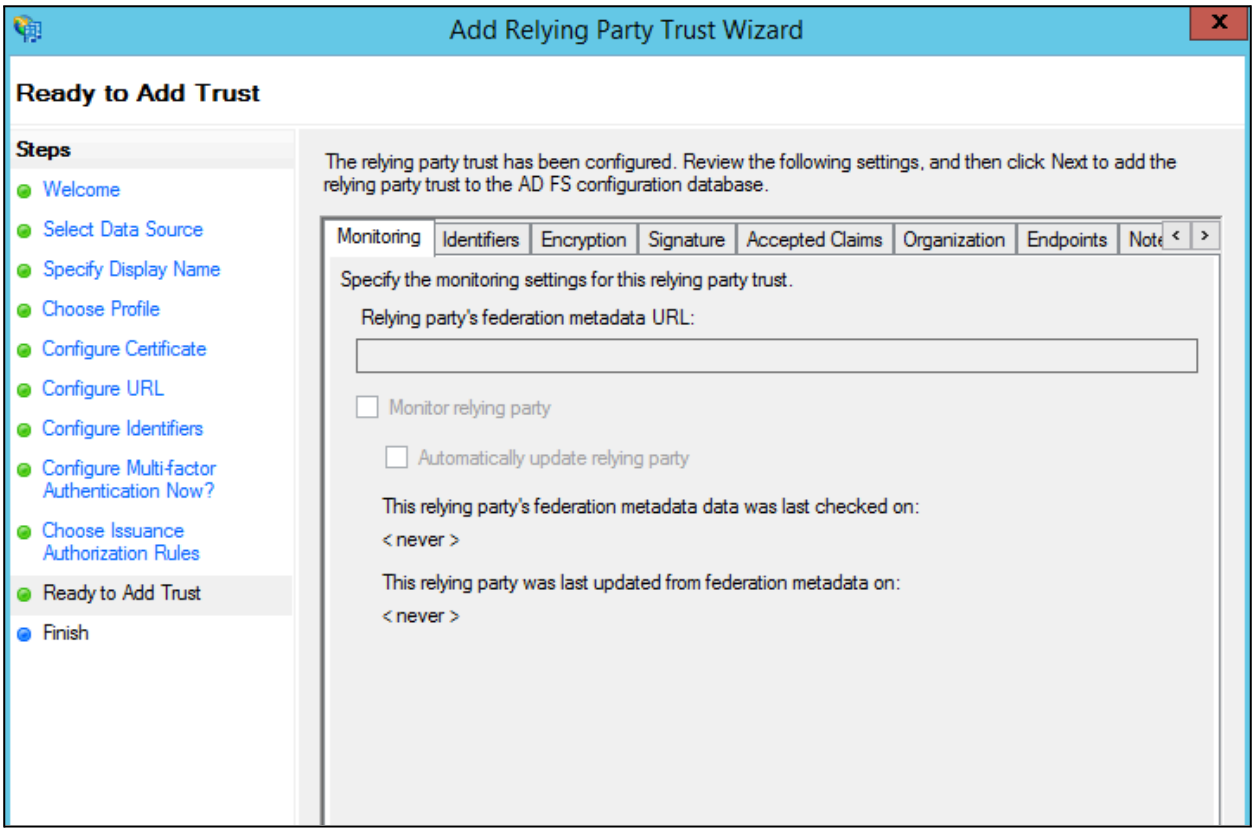


9. Allow all users to login.

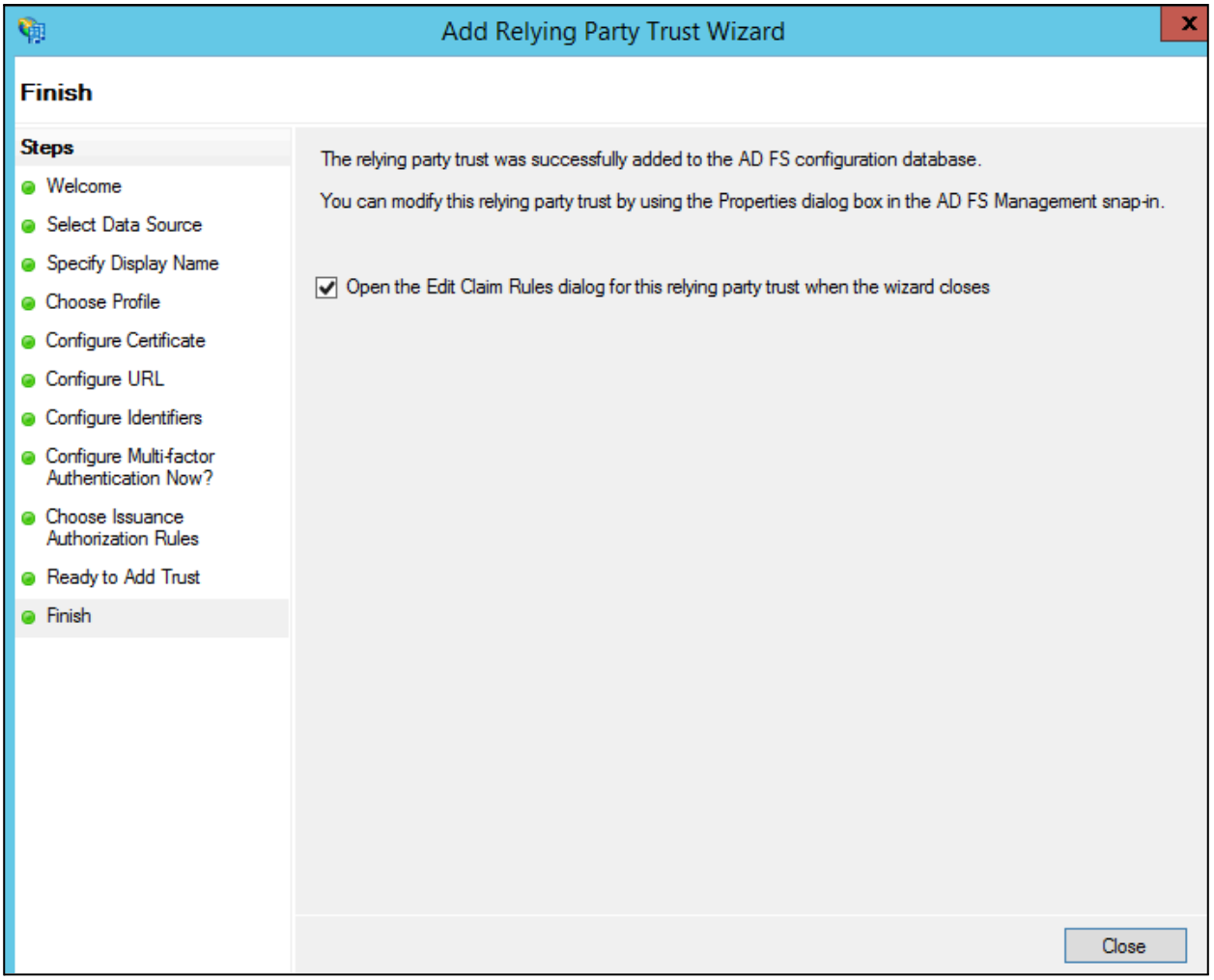
This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



10. Do not change anything here, just click next.



11. Close the “Relying Party” wizard



As you can see, the Relying Party is created, but right now there will be no claim that will be send to DNN, because there are no claim issues policy. Claim configuration is described in chapter below.

Claims configuration

The AD FS is using claims as a container to send Active Directory user profile fields to DNN. In following chapter we will define set of rules that defines which Active Directory user attributes needs to be send to DNN. The "ADFS-Pro Authentication" requires following claims:

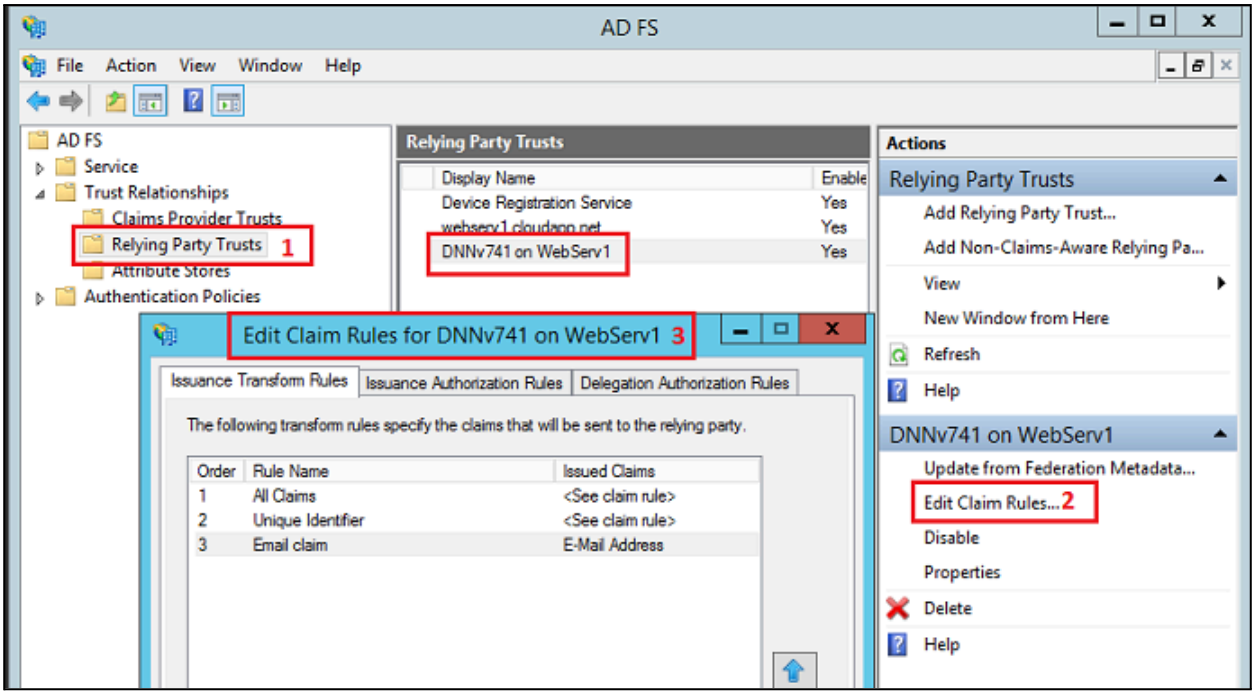
- Name identifier, claim type `http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name` described in rule #1,
- UPN claim, claim type `http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn` described in rule #2

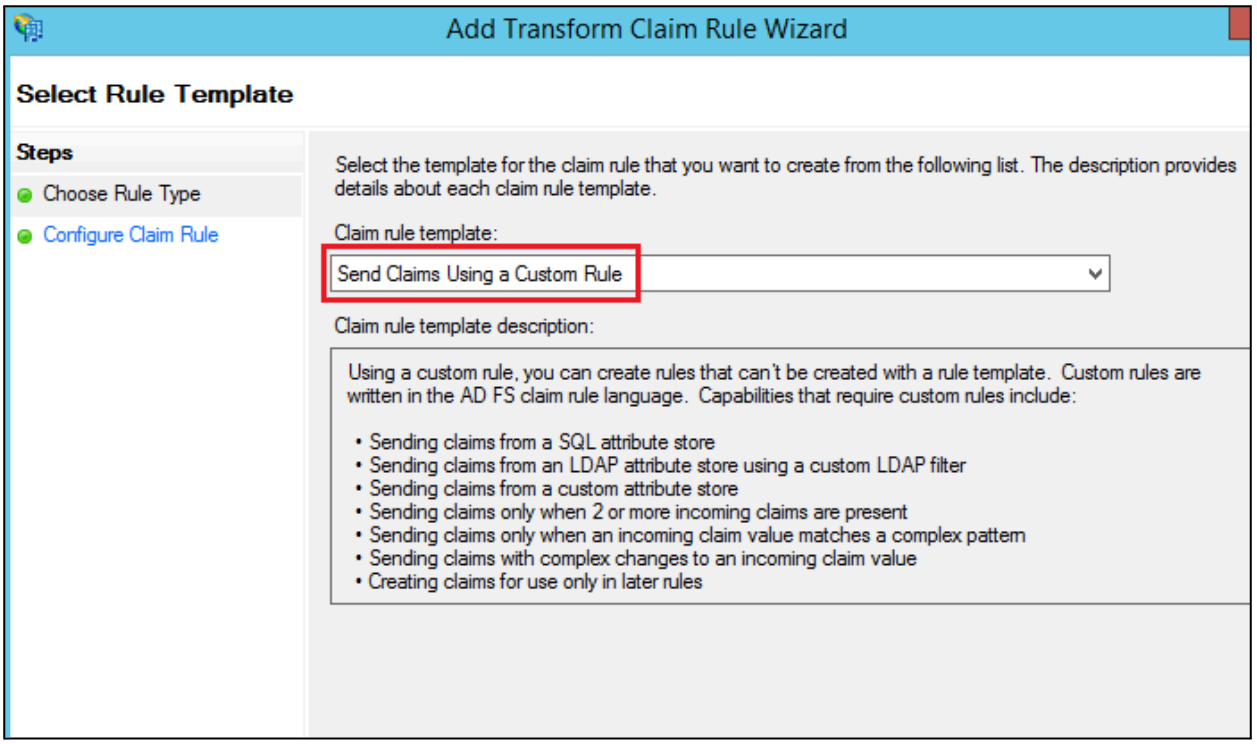
Claim #3 that contains AD user profile fields and claim #4 that has list of AD user groups are not mandatory.

Below are the steps that describe how to create these claims.

1. Select the "Relying Party" that was created above. Then click on the "Edit claim rules". Select claim rule template "Send claims using custom rule". See figures below.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html





First rule will populate the identity claim.

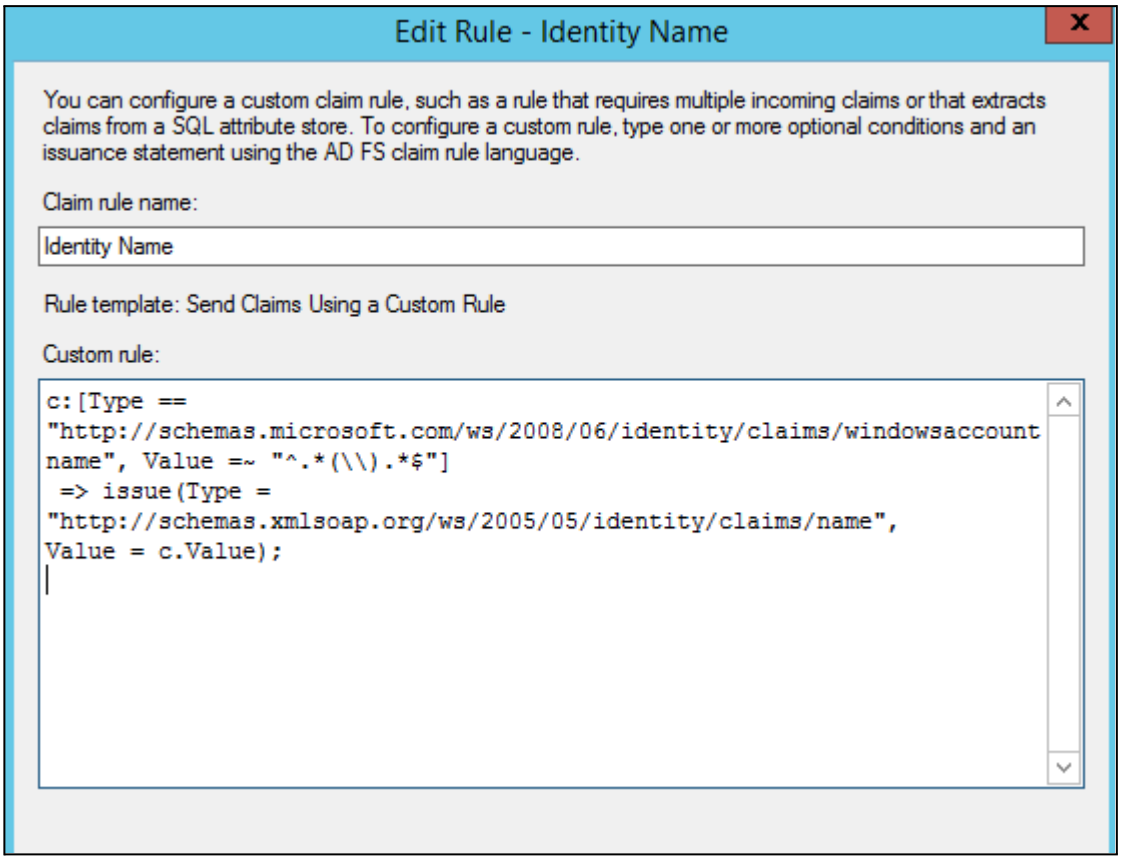
Please note that:
Before you will configure rule for the identity claim, you need to know two things:

- 'ADFS-Pro Authentication' offers storing usernames in multiple formats, see this chapter,
- value of the ADFS identity claim needs to be the same as the DNN username.

Below 'identity claim' rule works perfectly if DNN username is in format 'DomainName\Username'.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

```
c:[
Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccountname", Value =~
"^.*(\\).*$"]
=> issue(
Type = "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name",
Value = c.Value);
```



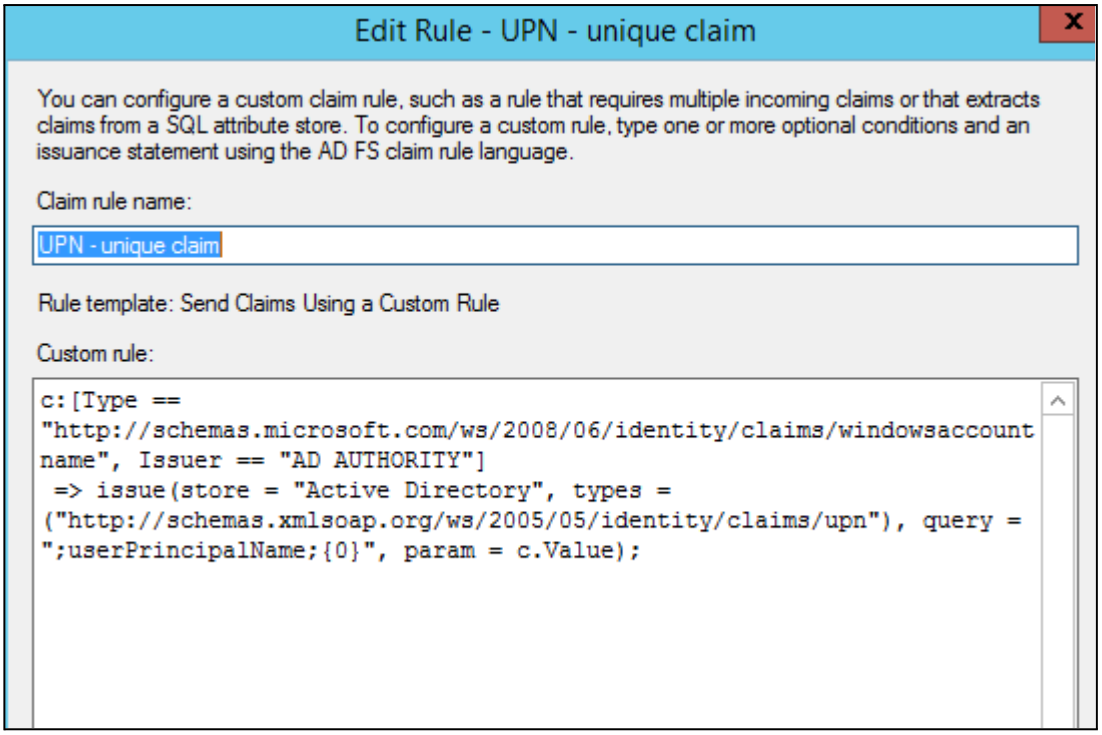
If DNN username will be saved in format 'Username' or 'Multiuser' that doesn't have the domain prefix, the ADFS identity claim rule will be as follow:

```
c:[
Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccountname",
Value =~ "^.*(\\).*$"]
=>
issue(Type="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name", Value=RegexReplace(c.Value,
".*\\", ""));
```

Second rule will output unique claim, by default it's upn:

```
c:[
Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccountname",
Issuer == "AD AUTHORITY"]
=>issue(
store = "Active Directory",
types = ("http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn"),
query = ";userPrincipalName;{0}", param = c.Value);
```

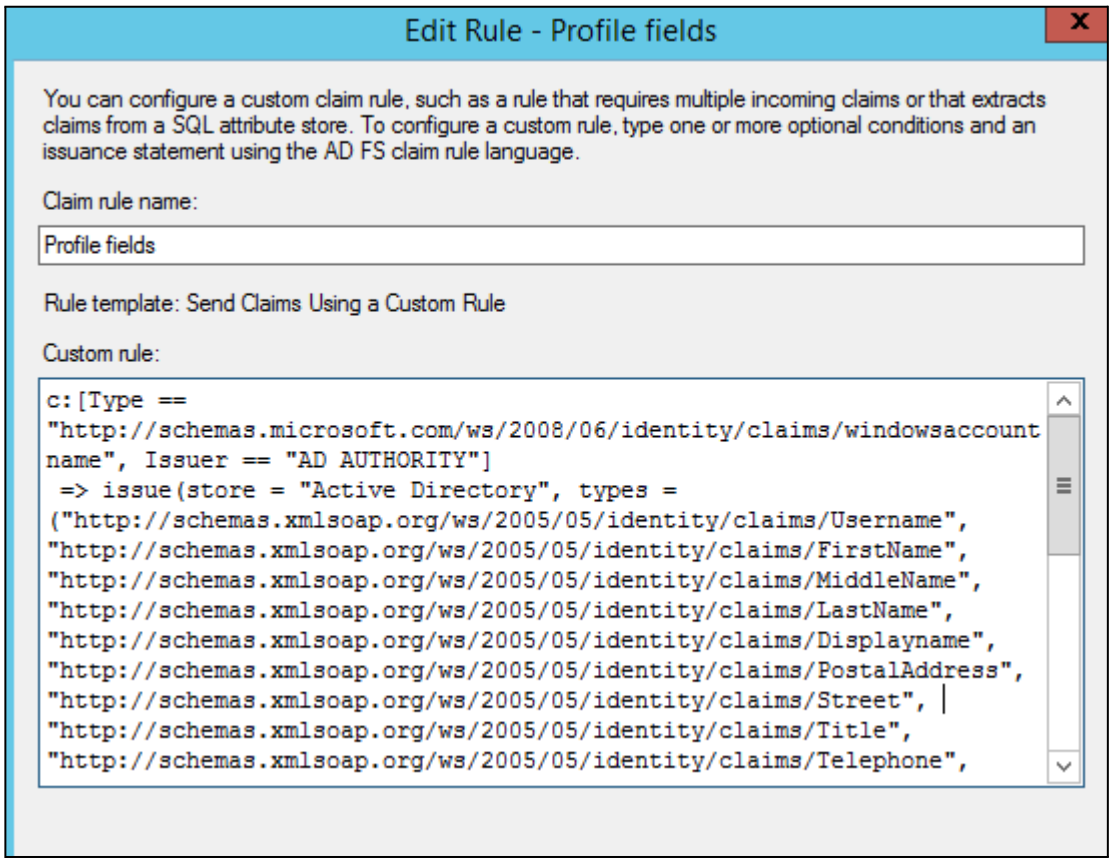
This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



Third rule will output number of claims that contains AD user profile fields:

```
c:[Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccountname",
Issuer == "AD AUTHORITY"]
 => issue(store = "Active Directory", types =
("http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Username",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/FirstName",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/MiddleName",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/LastName",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Displayname",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/PostalAddress",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Street",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Title",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Telephone",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Cell",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Fax",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Email",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/City",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Region",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Biography",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/PostalCode",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Office",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Department",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Company",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Website",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/ipPhone",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/Pager"), query =
";sAMAccountName,givenName,initials,sn,displayname,PostalAddress,StreetAddress,title,telephone
number,Mobile,FacsimileTelephoneNumber,mail,l,st,description,postalCode,physicalDeliveryOffice
Name,department,company,wwwhomepage,ipPhone,pager;{0}", param = c.Value);
```

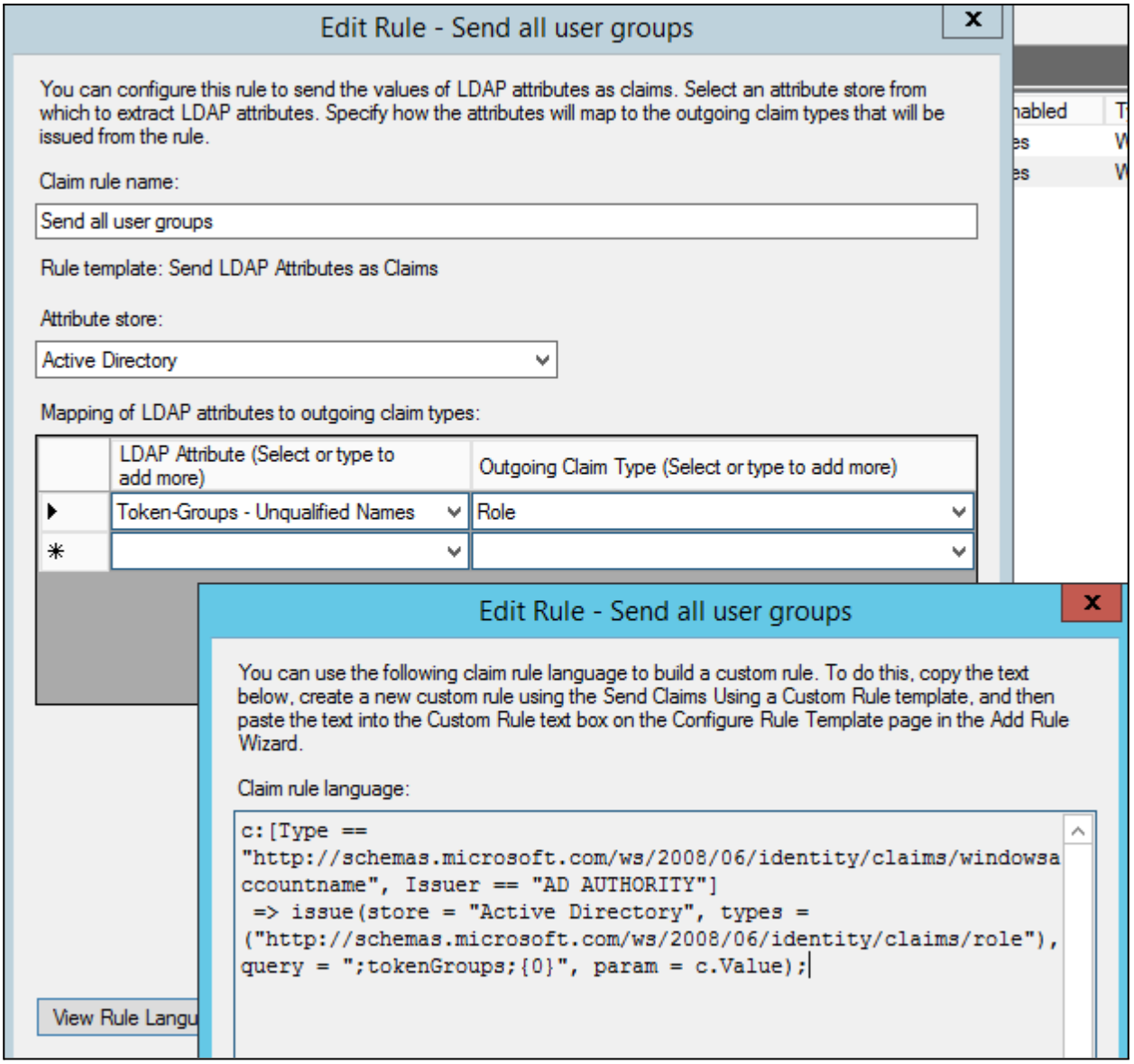
This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



Below is rule that can issue list of AD user groups:

```
c:[
Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccountname", Issuer
== "AD AUTHORITY"]
 => issue(
store = "Active Directory",
types = ("http://schemas.microsoft.com/ws/2008/06/identity/claims/role"),
query = ";tokenGroups;{0}",
param = c.Value);
```

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



Below are two rules that can issue only specific AD groups. For example to output only groups that contains string “Domain” in the group name (like: Domain Users), create following two rules.

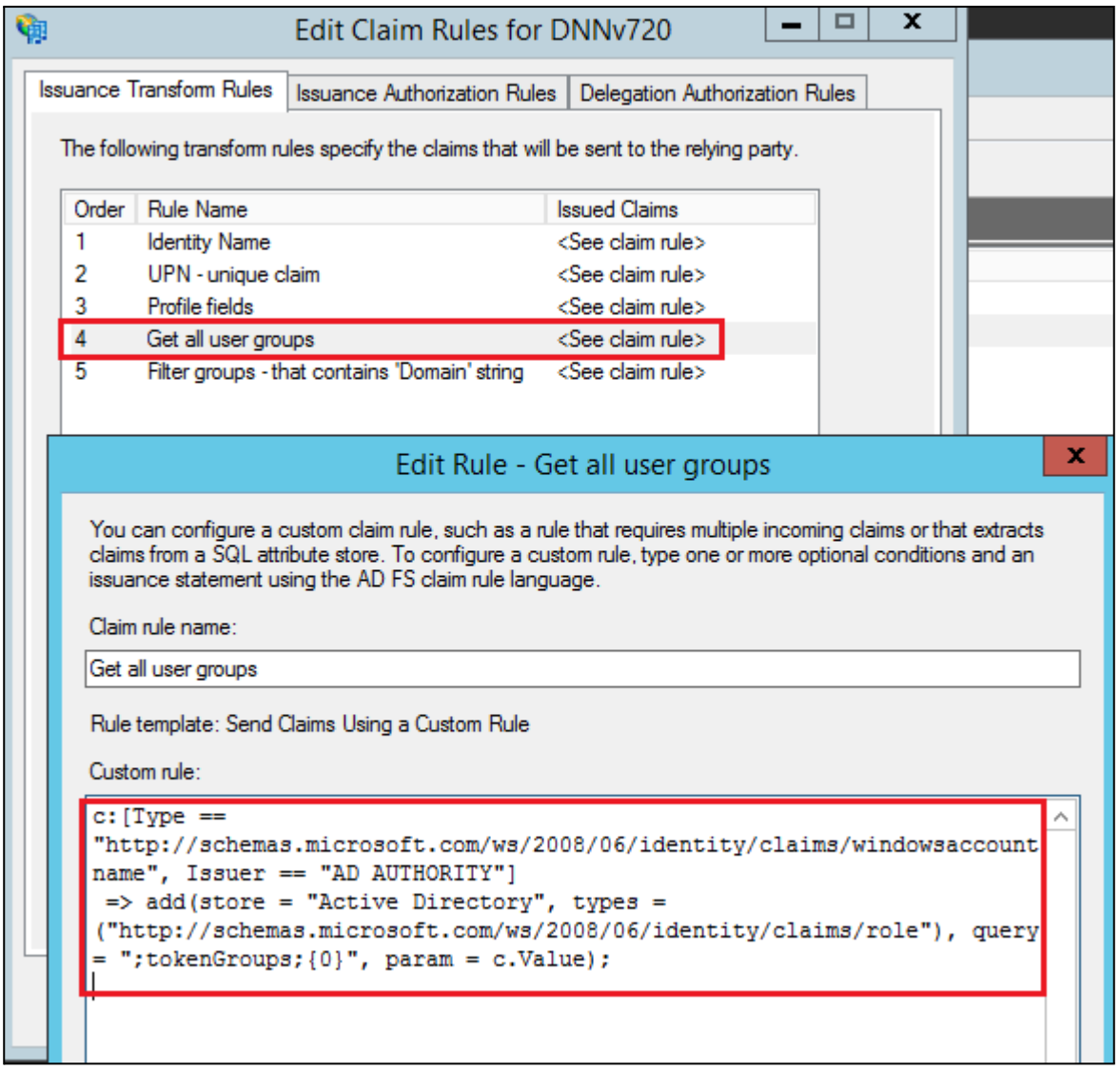
Rule 1 (on the figure below named “Get all user groups”)

```
c:[
Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccountname", Issuer
== "AD AUTHORITY"]
 => add(
store = "Active Directory",
types = ("http://schemas.microsoft.com/ws/2008/06/identity/claims/role"),
query = ";tokenGroups;{0}",
param = c.Value);
```

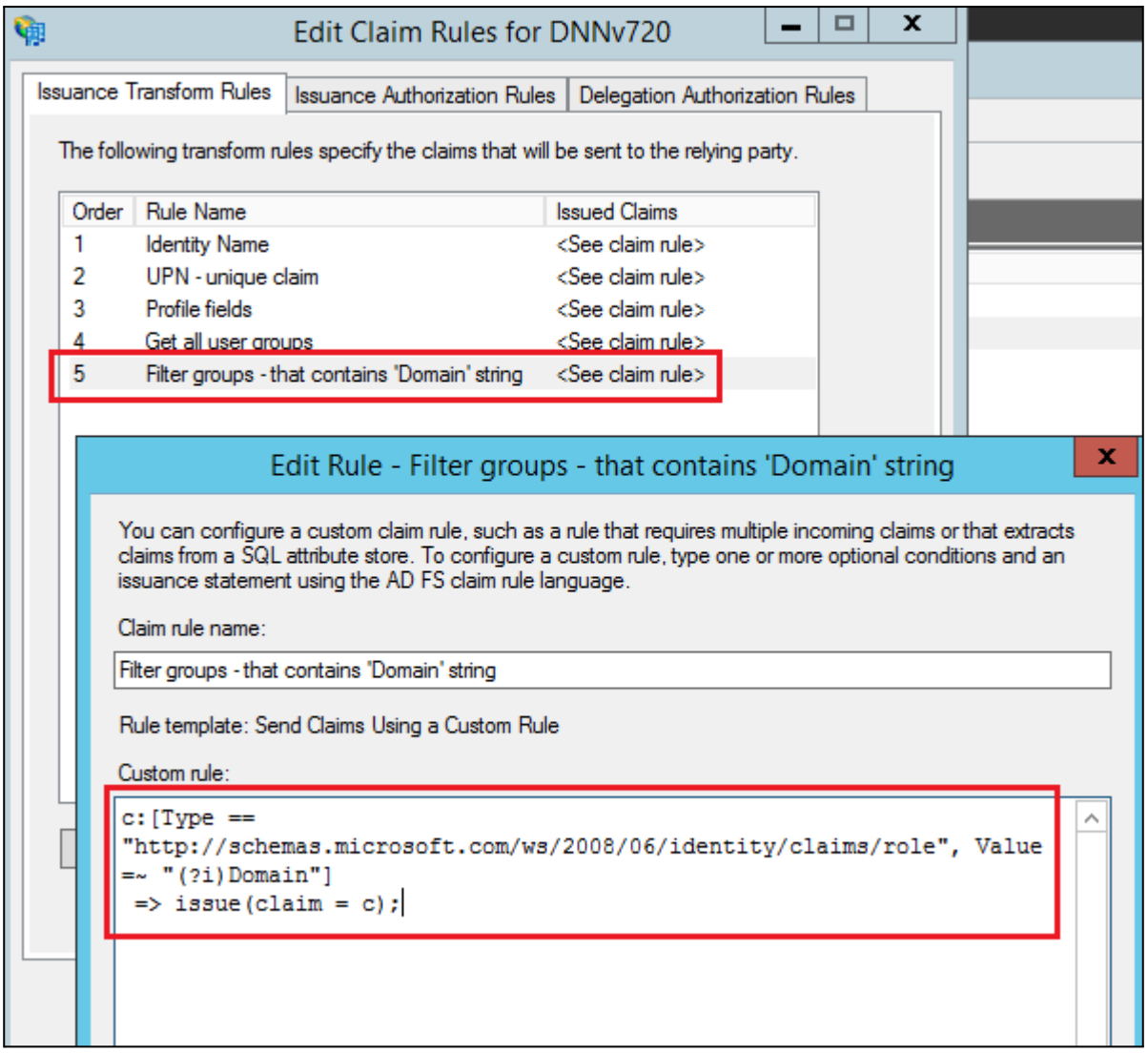
Rule 2 (on the figure below named “Filter groups- that contains ‘Domain’ string)

```
c:[
Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/role",
Value =~ "(?i)Domain"]
 => issue(claim = c);
```

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



Customizing AD FS login page

https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/dn280950(v=ws.11)?redirectedfrom=MSDN

DNN provider configuration

Overview

By default you don't need to change anything in the DNN settings. All the configuration tasks are related to the Glanton provider "ADFS-Pro Authentication". Please note that DNN needs to be configured with HTTPS protocol. The AD FS will work only with websites that are working over the HTTPS.

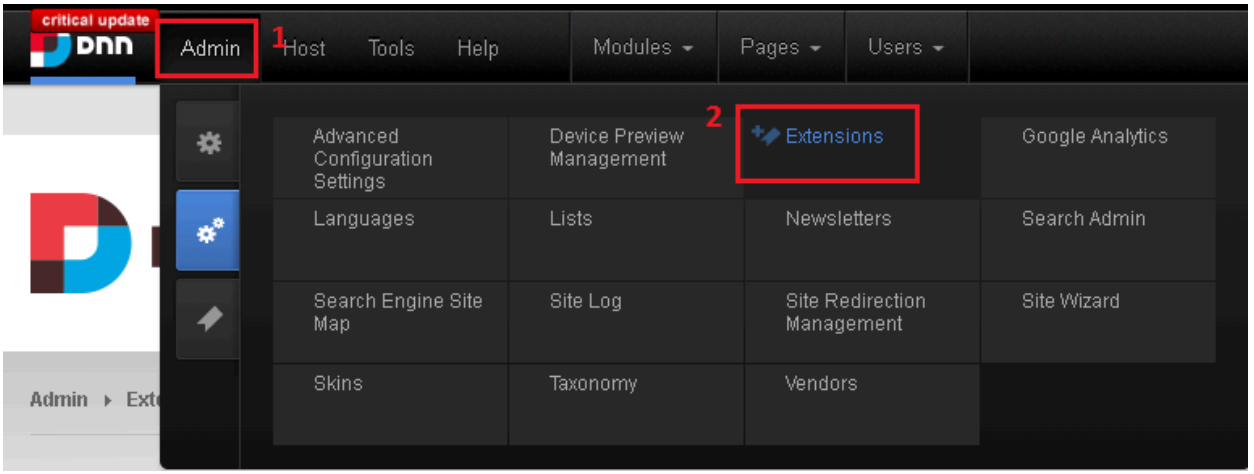
Provider installation

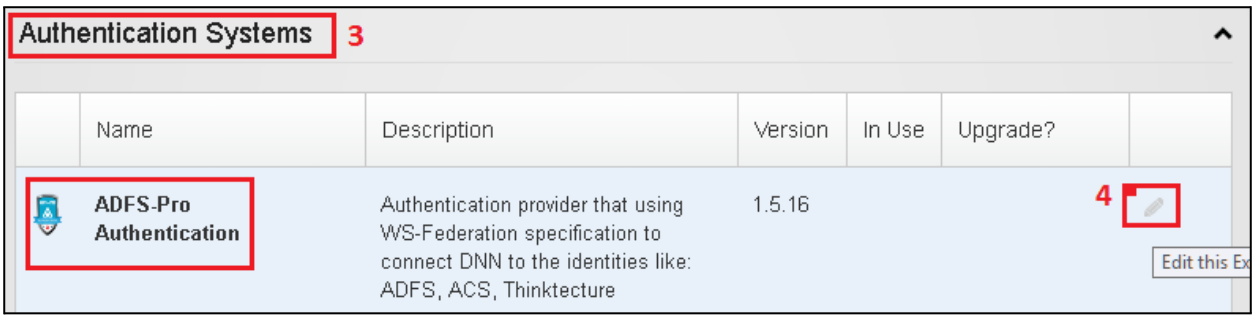
Install the "ADFS-Pro Authentication" extension in DNN. It's a standard installation procedure that can be initiated under the "Host->Extensions" menu.

Provider configuration

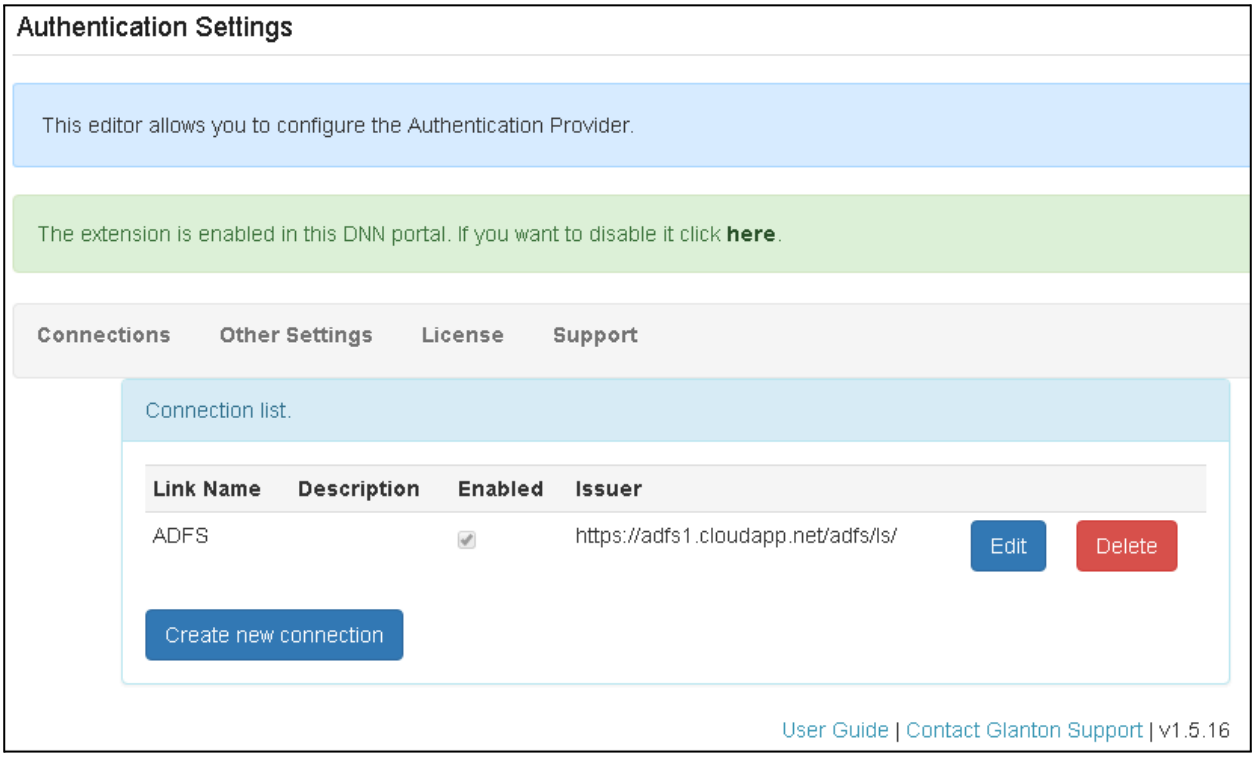
This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

To configure the “ADFS-Pro Authentication” you needs to open the settings menu. Provider settings are available for DNN host or DNN admin user. Provider settings are located under the “Admin-> Extensions-> Authentication Systems”. To open the settings click on the “Pencil” icon. See figure below.





After that should be displayed form like on the figure below.



This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

Provider activation

To activate the “ADFS-Pro Authentication” extension you need a “license key”. To create the “license key” you need:

- “install key”, that can be found in provider settings,
- “invoice number”, that can be found in email from DNN Store when the product was purchased;

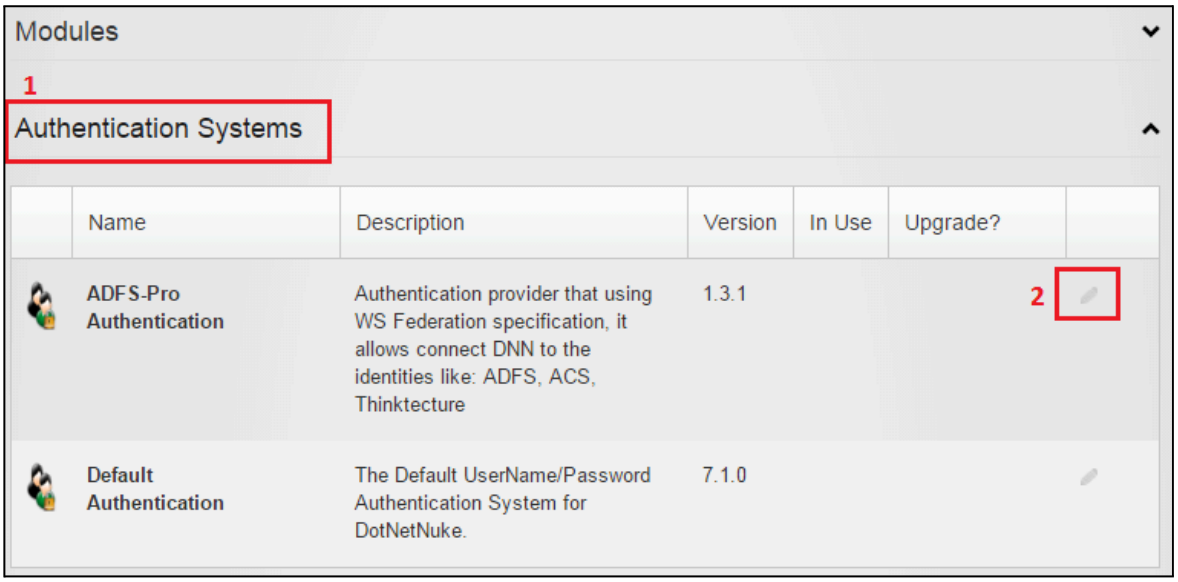
The “ADFS-Pro Authentication” module will work only with the correct license key. There are two kinds of licenses:

- 14 days free trial license,
- full paid license;

Both licenses are for multiple portals within one DNN installation/instance. When you purchase full license, you won't lose any of your trial settings.

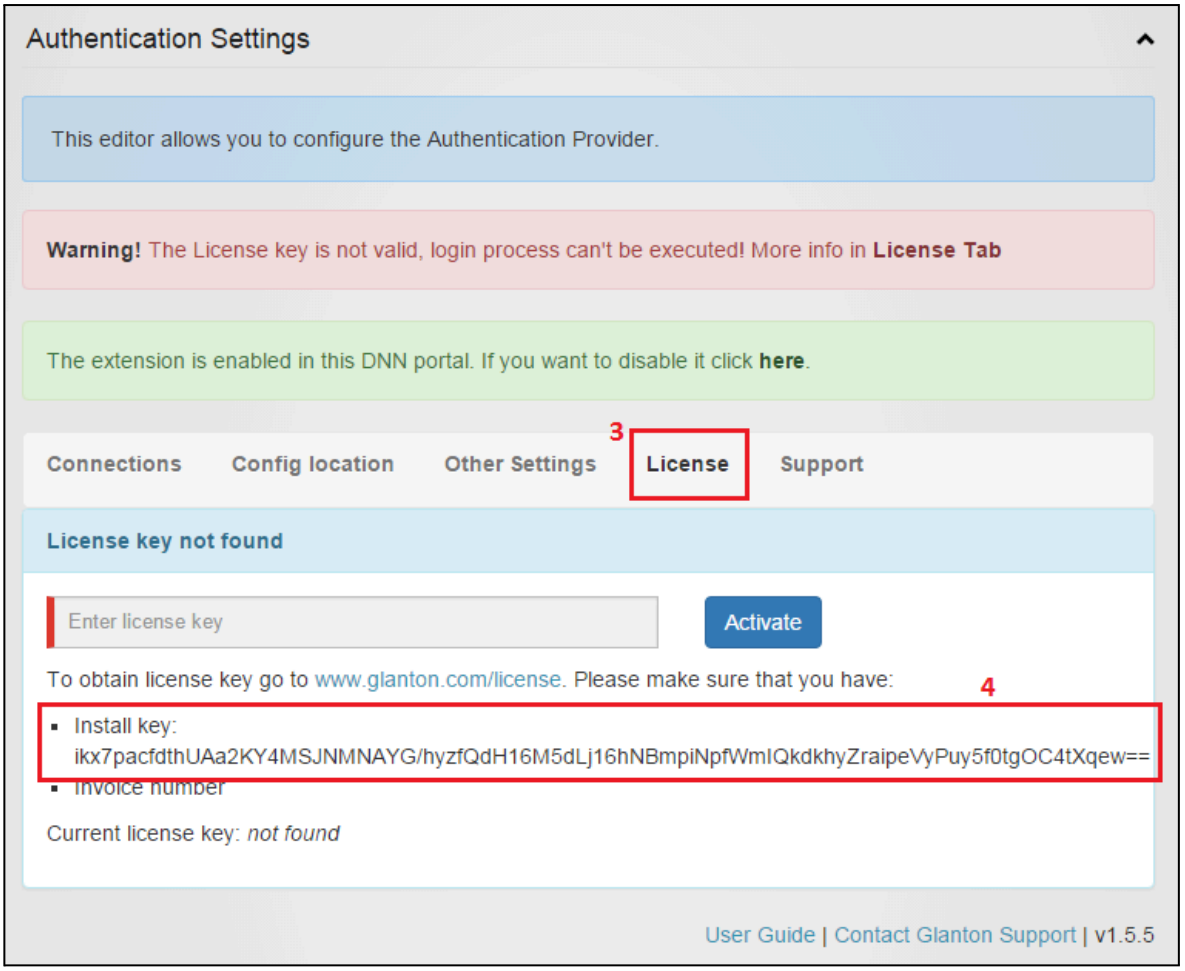
To get a license key, first you need a unique “install key”. To get the “install key” follow steps below:

- login to DNN as a Host user.
- go to Admin-> Extensions-> Authentication Systems, and click on the pencil near the “ADFS-Pro Authentication” extension, see figure below:



- click on “License” tab, the install key should be displayed at the bottom of the form, see figure below:

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



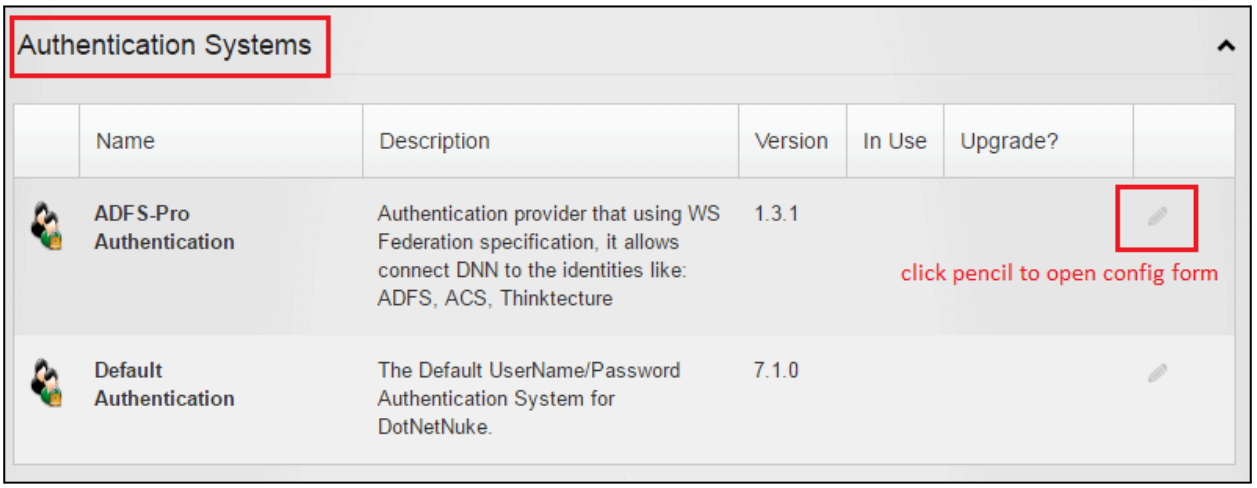
- When you have the "install key" and the invoice number please go to http://modules.glanton.com/license to create the "license key";

Note: if you don't have invoice number from DNN Store email: barry@glanton.com.

Enable (disable) provider

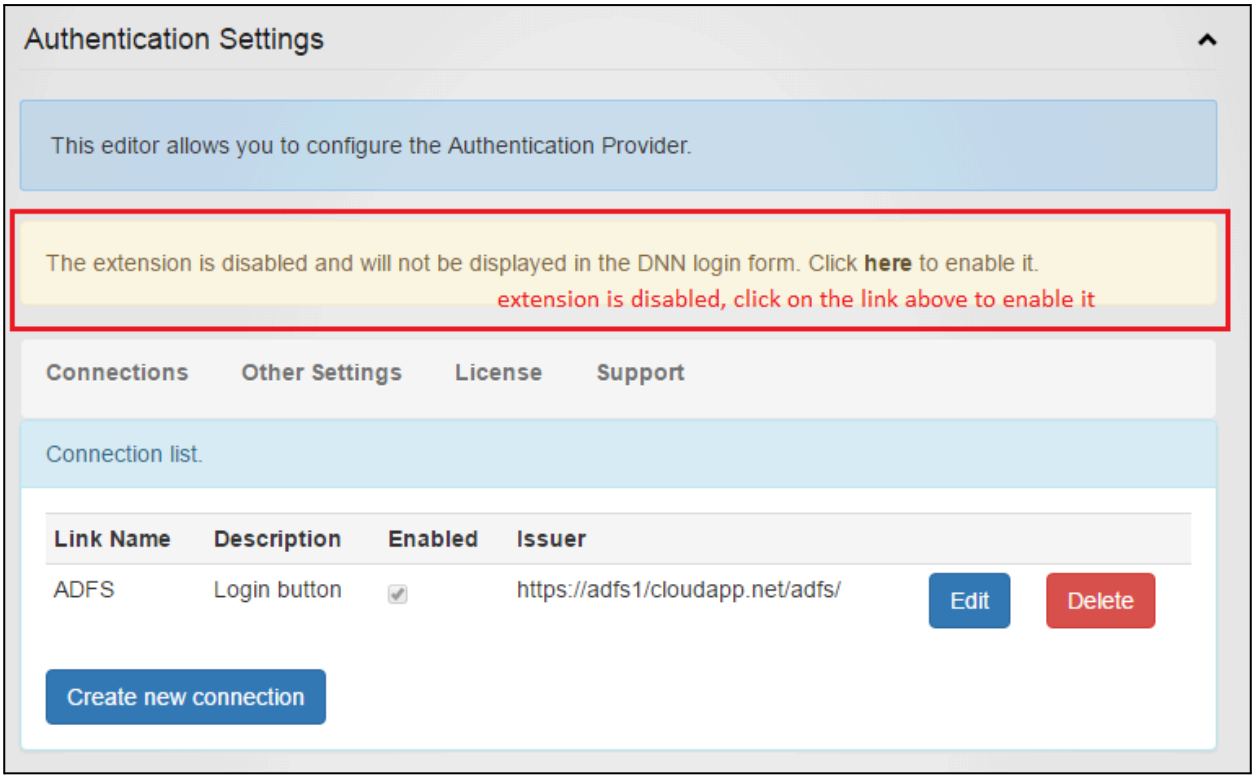
When "ADFS-Pro Authentication" is installed, before it will be used it must be enabled. This must be done for each DNN portal separately, it's a standard process for each authentication module. To enable "ADFS-Pro Authentication" please execute following steps:

1. Go to "Admin->Extensions->Authentication Systems". Next go to "ADFS-Pro Authentication" settings, by clicking on the pencil on the right. See figure below.

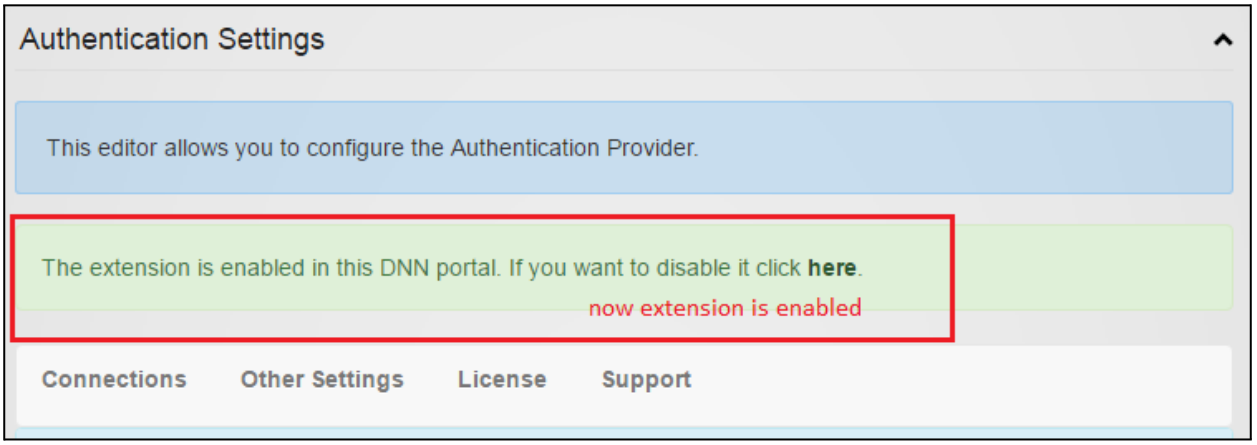


This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

2. On figure below message with yellow background inform you that extension is disabled. Click on the link *"here"* to enable it.

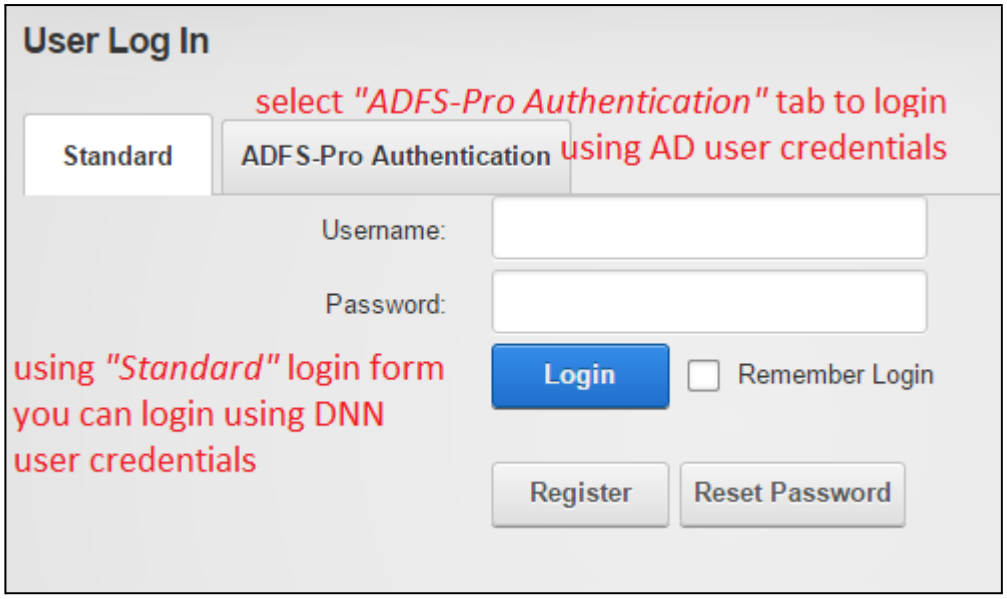


On figure below extension is enabled. If in the future you want to disable it, just click on the link *"here"*.



When "ADFS-Pro Authentication" extension is enabled, user can chose one of two login methods. First option "Standard" is for a core DNN users, like host, administrator etc. Second option "ADFS-Pro Authentication" is for Active Directory users. See figure below:

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

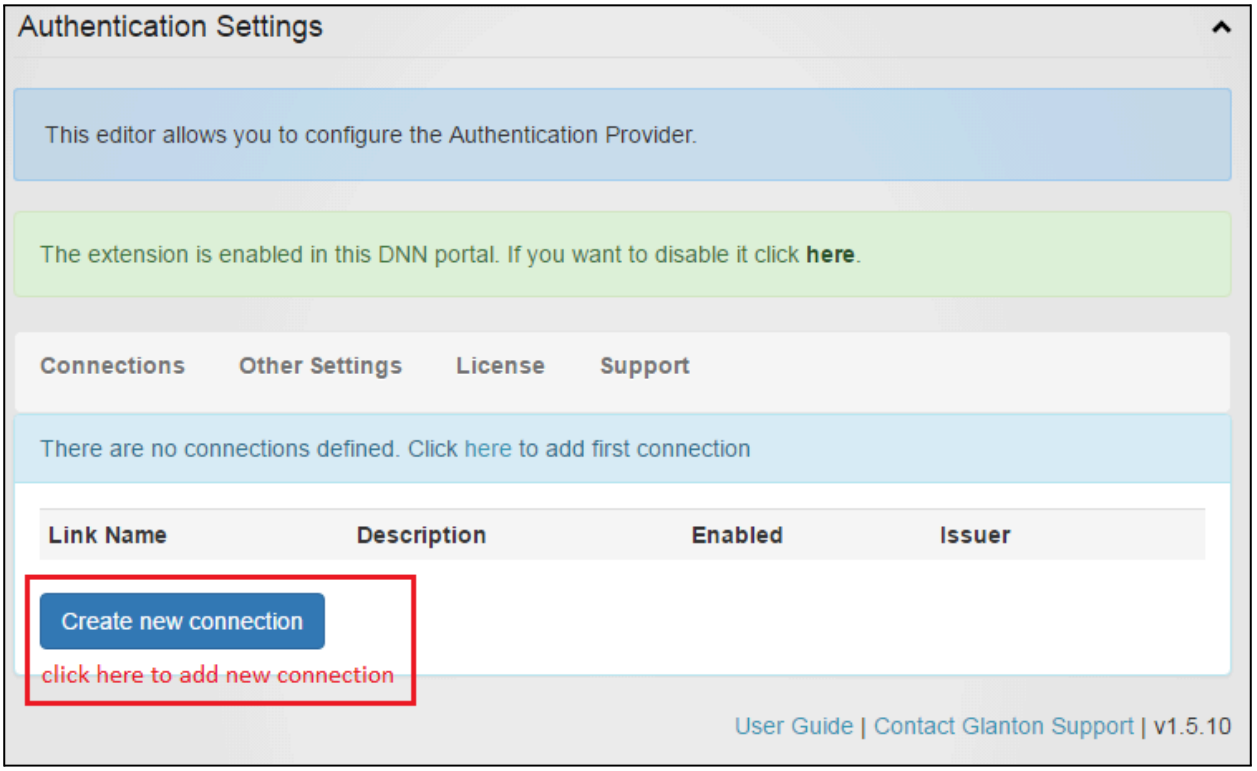


Creating connection between DNN and AD

To add DNN website to the AD FS a "ADFS-Pro Authentication" provider must be configured. I'm assuming that it's already installed in DNN and on AD FS side corresponding 'Relying Party" is created (see previous chapter).

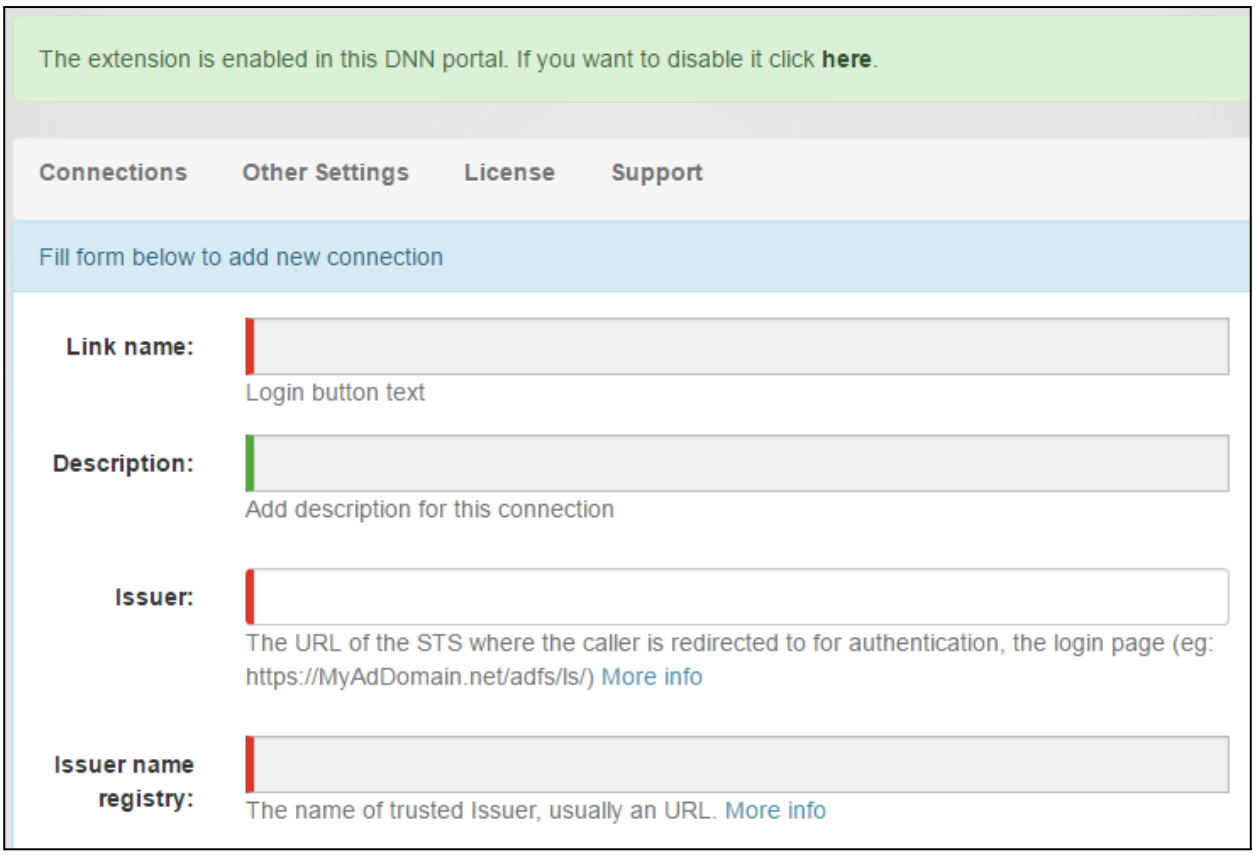
To create connection to ADFS follow steps below.

1. Sign in as DNN admin or host. Go to "Admin->Extensions->Authentication Systems". Next go to "ADFS-Pro Authentication" settings, by clicking on the pencil on the right.
2. Select button "Create new extension", see figure below.



3. On the screen below is a blank form that needs to be filled to establish communication with ADFS.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

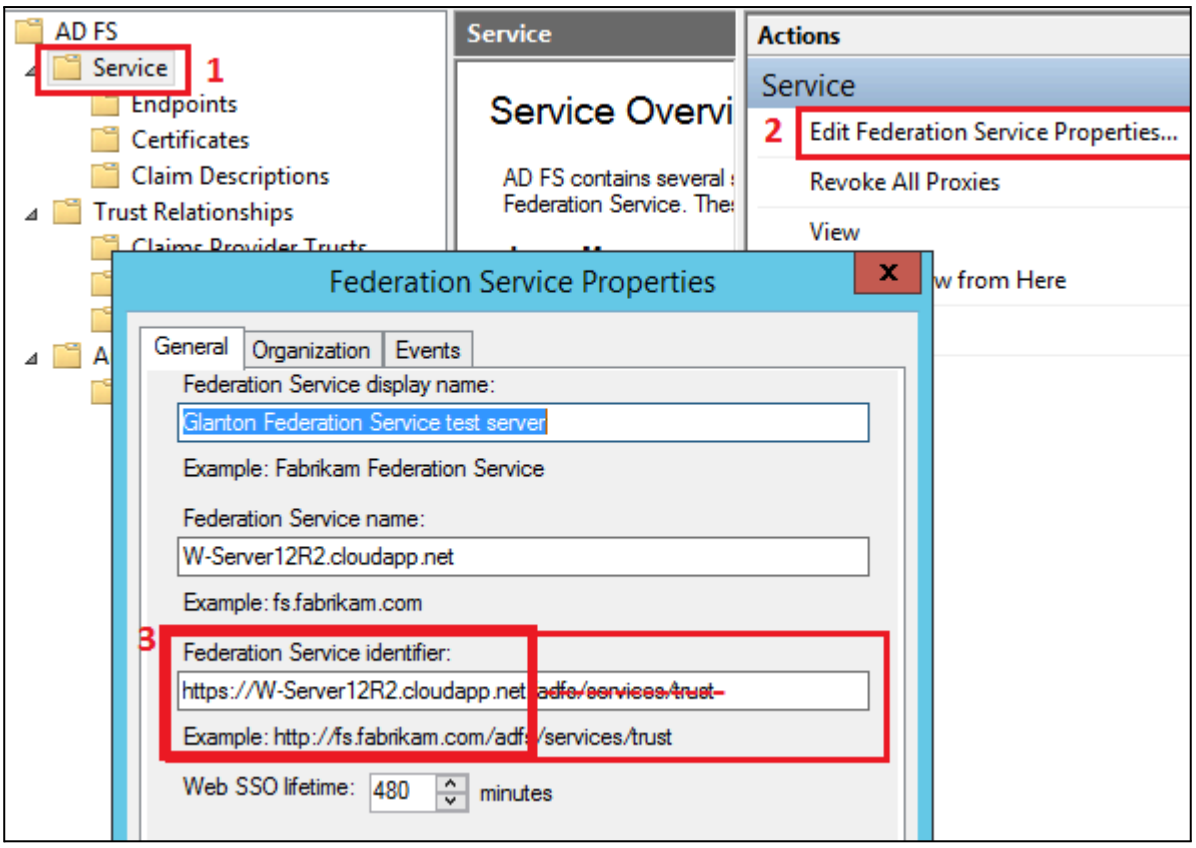


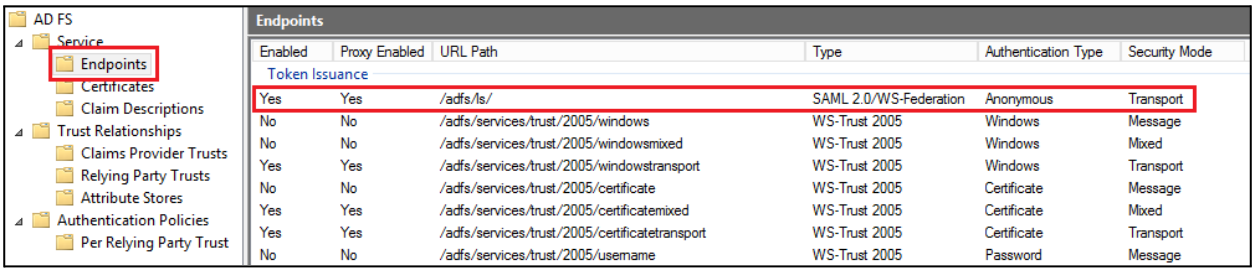
In the form above you must enter settings obtained from the AD FS. Below these settings are described.

Issuer

The "Issuer" property is a URL address of the security token service (STS), login service, to which to send WS-Federation sign-in and sign-out requests.
This URL usually starts with AD FS identifier and ends with "adfs/ls/". AD FS identifier can be found here:

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



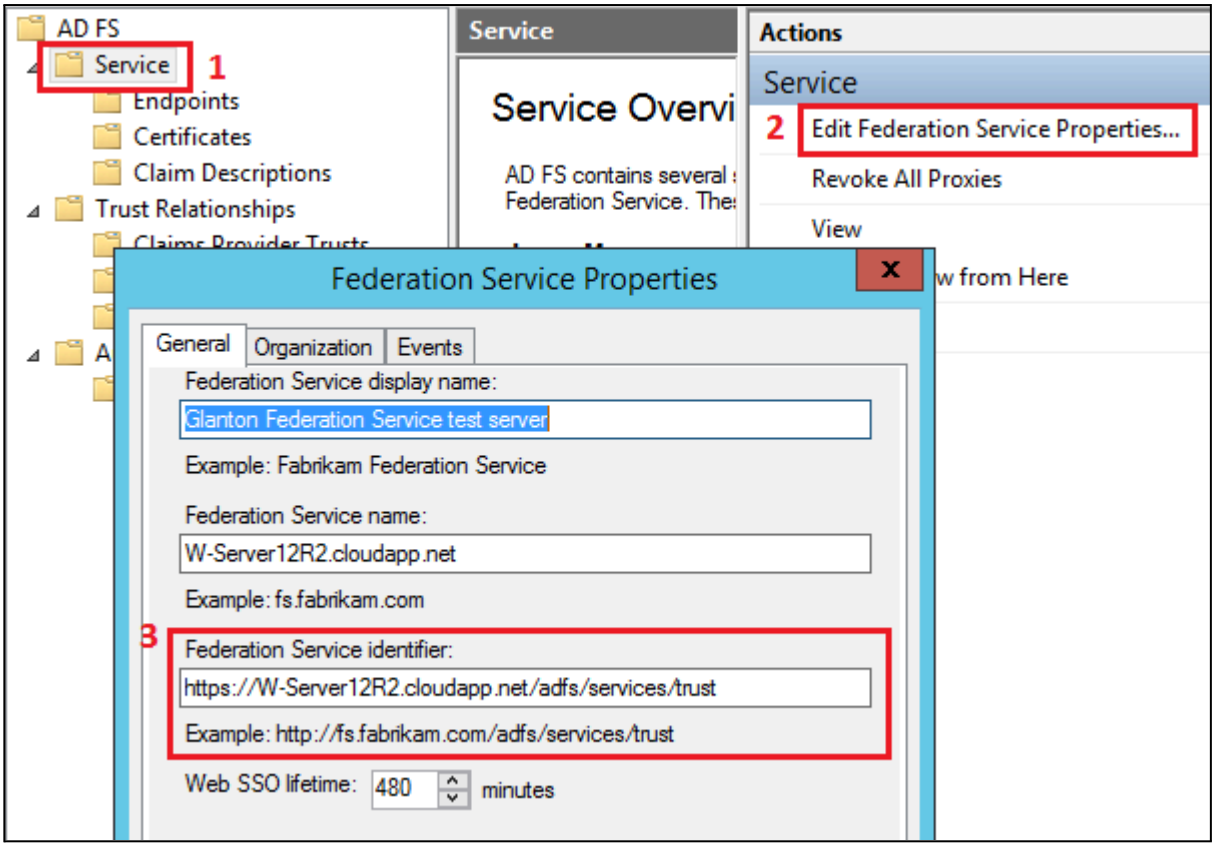


In our case issuer will be: `https://W-Server12R2.cloudapp.net/adfs/ls/`
Reference: https://msdn.microsoft.com...

Issuer Name Registry

"Issuer Name Registry" is a string (usually a url) that represents the Federation Service. To obtain the "Issuer Name Registry" for ADFS, follow steps from picture below:

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

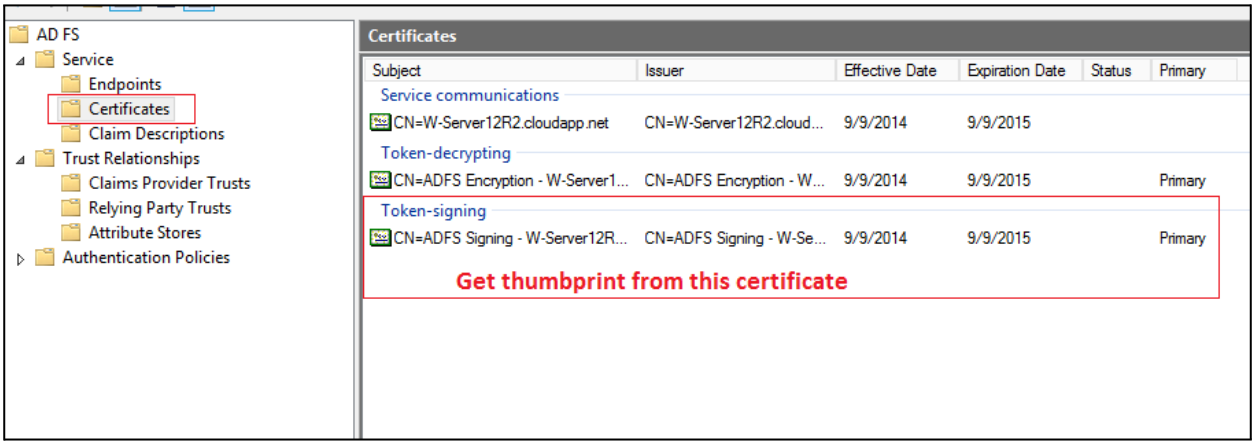


In our case "Issuer name registry" is: `https://W-Server12R2.cloudapp.net/adfs/services/trust`

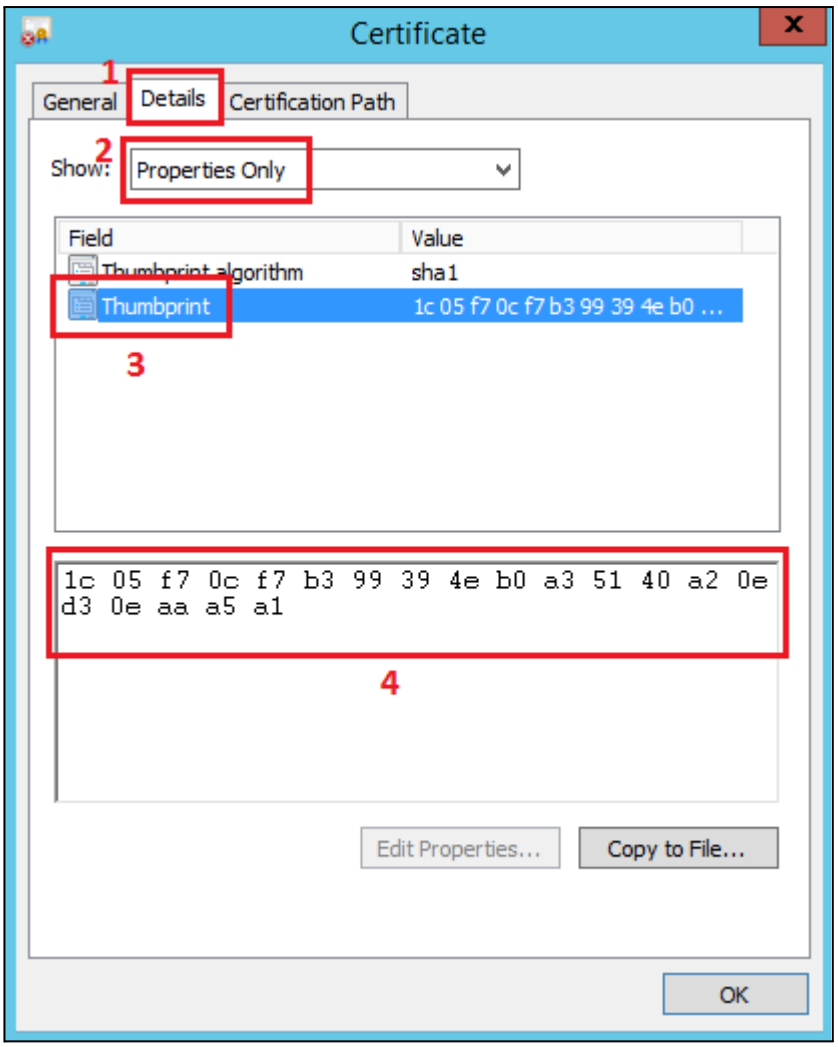
The "Issuer Name Registry" is associated to the certificate, both of these two factors allows to verify the signature of tokens produced by the issuer.

Certificate Thumbprint

You need a token signing certificate thumbprint, see figure below. Before you enter the value to the provider settings, remember to remove spaces from the thumbprint.



This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

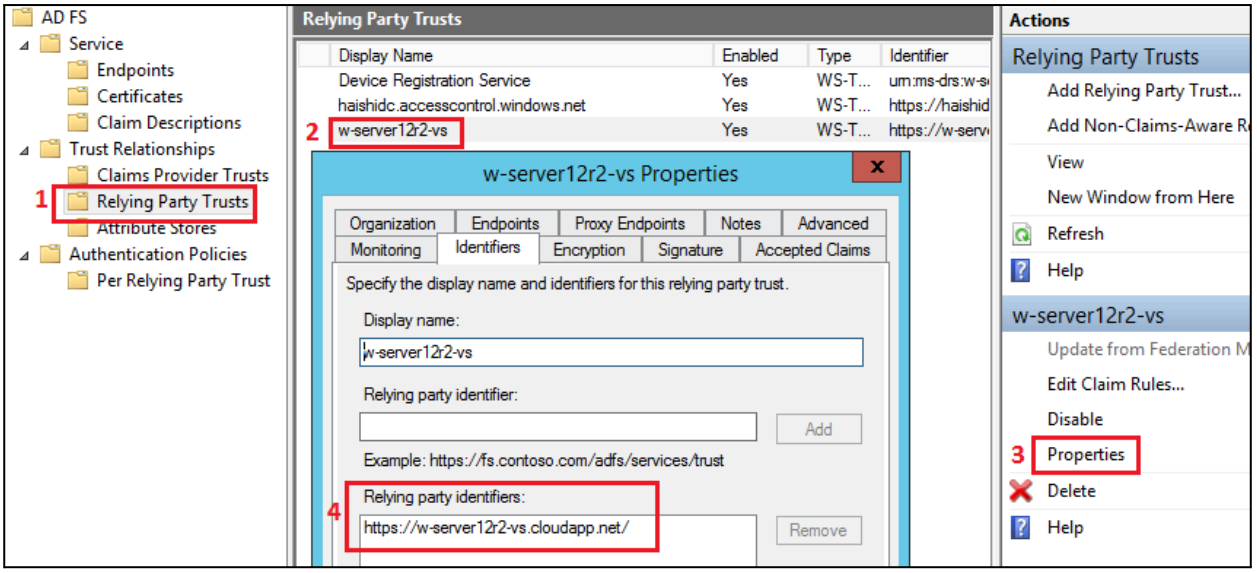


DNN provider must have thumbprint without spaces.

Realm

The realm is the identifier of your DNN application, usually it's a URL. It is used by the STS to know who we are. To obtain the "Realm" follow steps from the figure below. Usually it's a DNN website address. In our case "Realm" is: `https://w-server12r2-vs.cloudapp.net/`

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



Reference: https://msdn.microsoft.com/...

Home realm

The "Home Realm" is a identity provider (IP) address. By default "Home Realm" is equal to "Issuer". In the WS-Federation sign-in request "whr" parameter is equal to "Home realm". In our case it's: `https://W-Server12R2.cloudapp.net/adfs/ls/`

Audience Uri

"Audience URI" is an address (or a list of addresses) where user will back after sign in process. Usually it's a DNN website address, in our case "Audience Uri" is: `https://w-server12r2-vs.cloudapp.net/`
"Audience URI" will make sure that the token was really meant for our own DNN web application.

Authentication Type

The wauth parameter is like the wst:AuthenticationType element defined in WS-Trust. The value should be set to a string that contains a URI that identifies the type of authentication that is used.
The wauth parameter is optional. Set the AuthenticationType property to null or an empty string to remove the wauth parameter from the message.

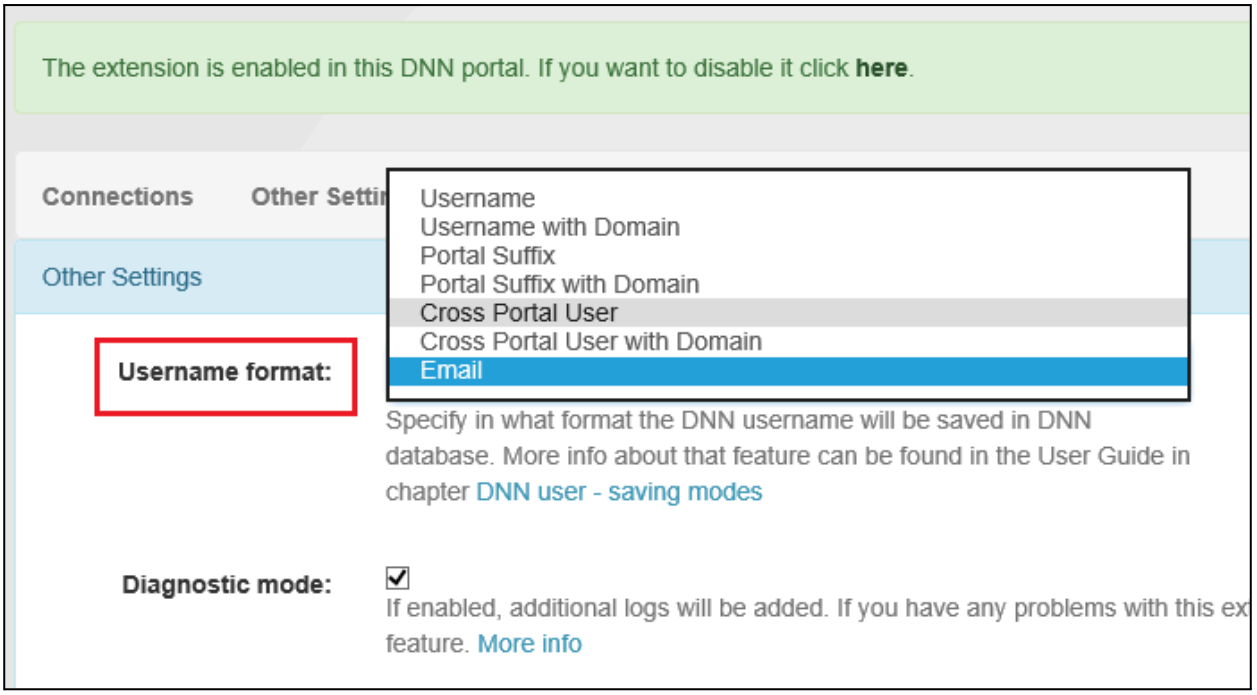
Passive Redirect Enabled

Specifies whether the WSFAM is enabled to automatically redirect unauthorized requests to an STS. Optional. The default is "true", unauthorized requests are automatically redirected.

DNN username formats

The "ADFS-Pro Authentication" module allows store DNN users in multiple formats. How the DNN username will be saved in DNN database in table "Users", can be specified in "Module Options-> Other Settings".

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



To choose the right username format you need to answer a few questions:

- how many portals exist in your DNN,
- do you want to allow to login one AD user to only one DNN portal or to each DNN portal,
- what is the existing DNN username format;

Below is a table that describes available DNN username formats, for the following assumptions:

- AD username is `Barry`,
- AD domain name is `GS`,
- DNN portal id is `2`;

Username format	Output example [for AD user: `Barry`, portal id: `2`, AD domain: `GS`]
Default `username`	"Default" username format allows save one Active Directory user across whole DNN install. In this situation AD user Barry is able to login to only one DNN portal (portal id = 2). output username: `Barry`
Default with Domain `Domain\username`	"Default with domain" username format allows save one Active Directory user across whole DNN install. In this situation AD user Barry is able to login to only one DNN portal (portal id = 2). output username: `GS\Barry`
Portal Suffix `username_{portal id}`	"Portal Suffix" username format allows save one Active Directory user in each DNN portal and it will be separate user instance. In fact every DNN portal contains his own DNN user, that points to

	one Active Directory user. In this situation AD user Barry is able to login to all DNN portals. output username: `Barry_2`
Portal Suffix with Domain `Domain\username_{portal ID}`	Active Directory user Barry can exist in each portal and it will be separate user instance. In fact every DNN portal contains his own DNN user, that points to one Active Directory user. In this situation AD user Barry is able to login to all DNN portals. output username: `GS\Barry_2`
Cross Portal User `username`	Active Directory user can exist in each DNN portal, his username will be the same, but with independent user profile. AD user Barry is able to login to all DNN portals. To enable this mode all "ADFS-Pro Authentication" instances across DNN install, should have "Username format" set to "Cross portal User". More info about the "Multi User" feature that allows to re-use username, can be found at this location: http://www.dnnsoftware.com/wiki/page/Users-in-multiple-portals-in-a-single-DNN-Instance output username: `Barry`
Cross Portal User with Domain `Domain\username`	Active Directory user can exist in each DNN portal, his username will be the same, but with independent user profile. AD user Barry is able to login to all DNN portals. To enable this mode all "ADFS-Pro Authentication" instances across DNN install, should have "Username format" set to "Cross portal User". More info about the "Multi User" feature that allows to re-use username, can be found at this location: http://www.dnnsoftware.com/wiki/page/Users-in-multiple-portals-in-a-single-DNN-Instance output username: GS\`Barry`
Email `username@domain.com`	"Email" username format allows save one Active Directory user across whole DNN install. In this situation AD user Barry is able to login to only one DNN portal (portal id = 2). output username: `Barry@gs.local`

Note: if username format is set to 'Username' or 'Multiuser', then 'Identity.Name' should be in the same format. Without the 'domain' name at the beginning as it is by default 'DomainName\Username'. Change first claim rule by following claim:

```
c:[
Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccountname",
Value =~ "^.*(\\).*$"]
=>
issue(Type="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name",        Value=RegexReplace(c.Value,
".*\\", ""));
```

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

Session token encryption

Session token allows the user to continue to browse to other pages within the same DNN application without having to re-authenticate with the identity provider for each page visit. This token is stored in the cookie that is encrypted. By default to write and read token in that cookie DPAPI (Data Protection API) is used. All works fine if we host our application in a single machine. Problem starts when you deploy your application to machine instances behind a load balancer, like an Azure. You can get an exception like:

A CryptographicException occurred when attempting to encrypt the cookie

More info in chapter "The data protection operation was unsuccessful"

To bypass that issue you must implement custom "session token" encryption. The idea is to use a certificate to encrypt and decrypt the session token. To do that add following lines in your web.config file:

```
<configuration>

<!-- the web.config code goes here -->

<!-- begin custom code -->
<system.identityModel.services>
    <federationConfiguration>
      <serviceCertificate>
     <certificateReference     x509FindType="FindByThumbprint"     findValue="ThumbPrintValue"
storeLocation="LocalMachine" storeName="My"/>
      </serviceCertificate>
    </federationConfiguration>
  </system.identityModel.services>
<!-- end custom code -->
</configuration>
```

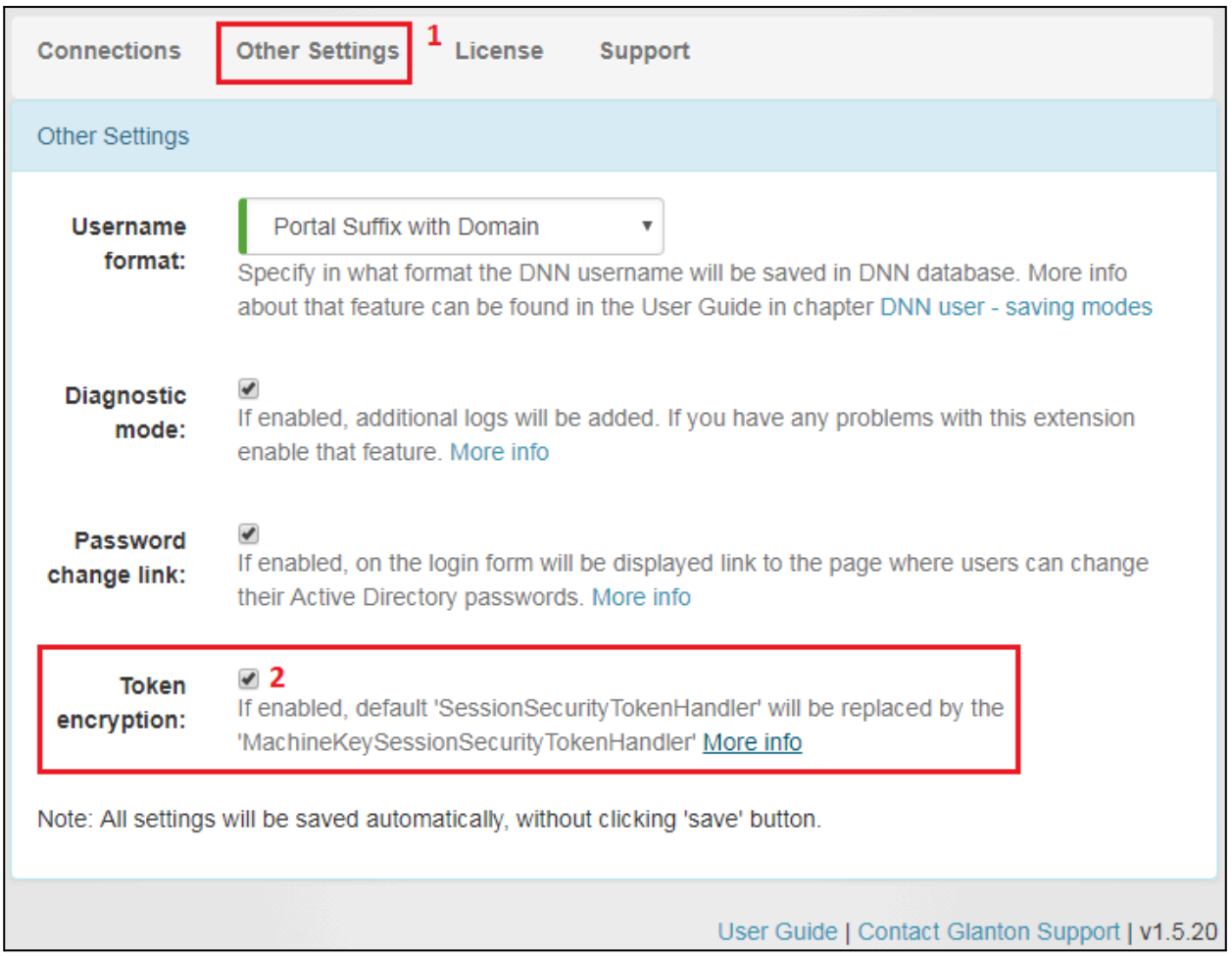
```
  <system.identityModel.services>
    <federationConfiguration>
      <serviceCertificate>
        <certificateReference x509FindType="FindByThumbprint"
                              findValue="9287292B3BFB90DF06895260D02C0EF5D55581C5"
                              storeLocation="LocalMachine" storeName="My"/>
      </serviceCertificate>
    </federationConfiguration>
  </system.identityModel.services>

</configuration>
```

Session Tokens protected by Machine Key

Session tokens by default, are protected with DPAPI which auto-generates a key that is specific to the machine. This means, by default, that session tokens won't work in a web farm or cloud based hosting. In that case session tokens can be configured to use the ASP.NET <machineKey> for protection instead.

To enable that option go to "ADFS-Pro Authentication-> Other Settings" and enable attribute "Token Encryption", see figure below:

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



References:
https://brockallen.com/2013/02/18/configuring-machine-key-protection-of-session-tokens-in-wif-and-thinktecture-identitymodel/
https://brockallen.com/2013/02/14/configuring-session-token-lifetime-in-wif-with-the-session-authentication-module-sam-and-thinktecture-identitymodel/

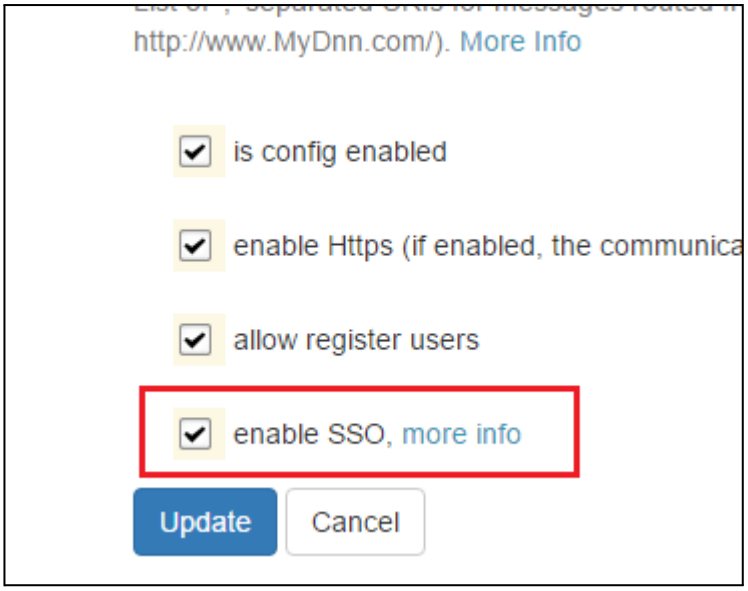
Single Sign On

Following conditions must be meet for SSO:

- The Web-proxy configured on the client should be configured to bypass proxy, for request to ADFS URL
- The ADFS URL (eg. http://MyDnnWebsite.com) should be added to the IE > Security >Intranet zones > Site list

Module offers single sign on. When user clicks on “login” link module will automatically redirect user to the ADFS login page. All you need to do is to go to: Admin-> Extensions-> ADFS-Pro Authentication, edit the config, and enable “enable SSO” attribute.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



If the attribute "enable SSO" is enabled, all anonymous users will be redirected to the STS login page (ADFS login page). To allow login for non AD users like DNN host or DNN admin, please add to the login page query string parameter: `?sso=false`. This query string parameter will stop "auto redirect" process.
If this doesn't help, delete the module settings using this SQL command:
TRUNCATE TABLE GS_FP_FederationConfig

Colleagues only enter their credentials once; as long as these credentials remain valid, the website or web service can be accessed without the need to manually log on.

Password change

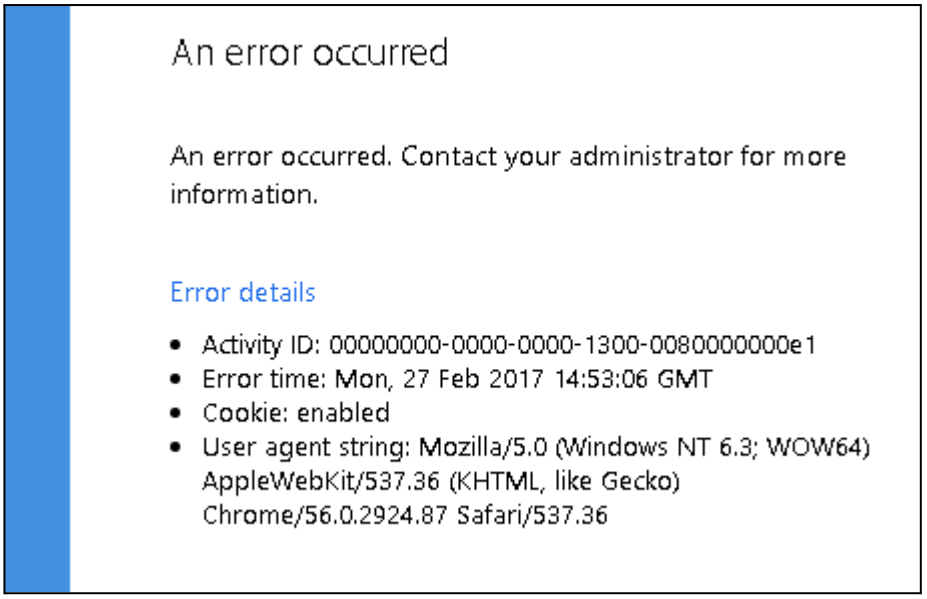
Background

Password change is the ability for a user to change his password with knowledge of the old password. Note that this is not password reset where the user does not know his password. This feature is usually for employees that are not connected to the corporate network where they have "self reset passwords" at the press of Ctrl+Alt+Del keys.

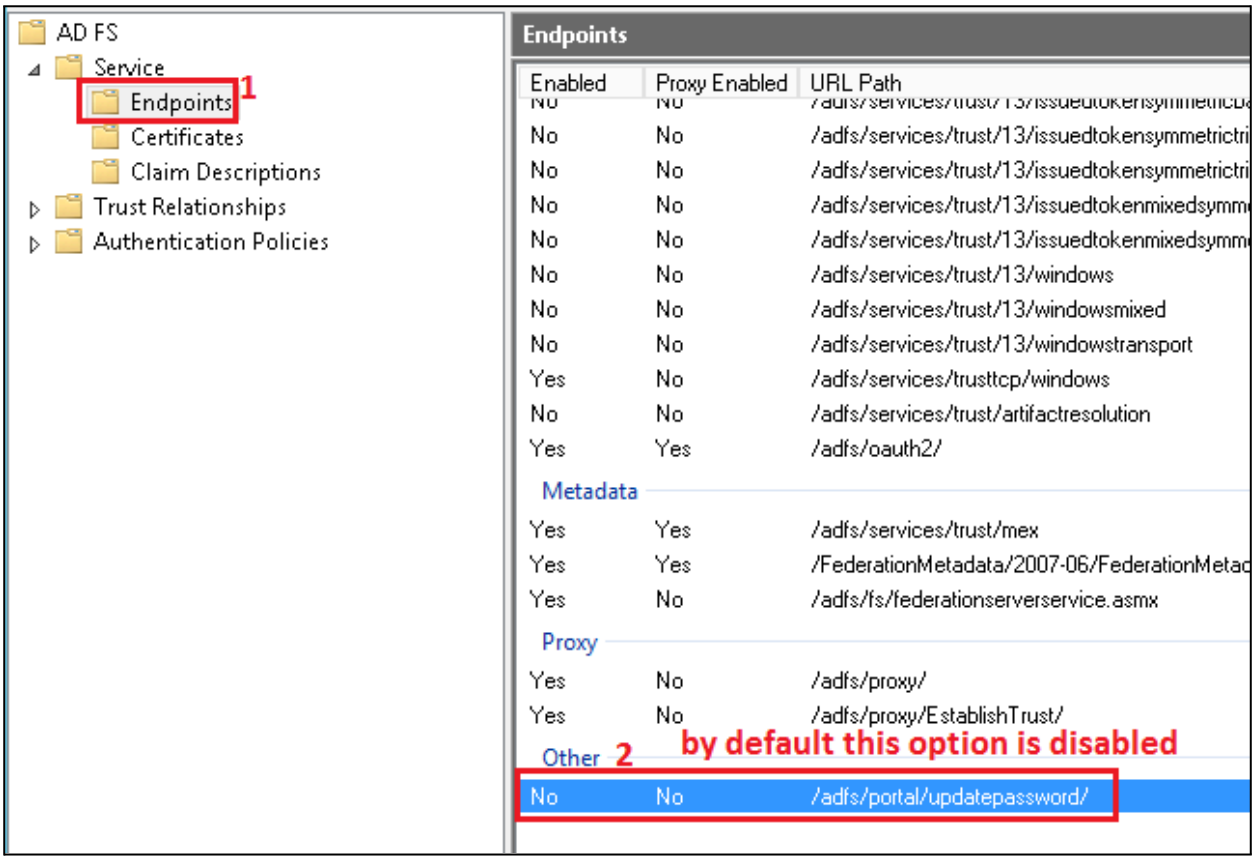
ADFS configuration

ADFS provides a special page to change user password that is available at following address:
https://YourAdfsServer.com/adfs/portal/updatepassword/
But for security reason '/adfs/portal/updatepassword/' endpoint is by default disabled and when you try to open page you will get message like on the screen below.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

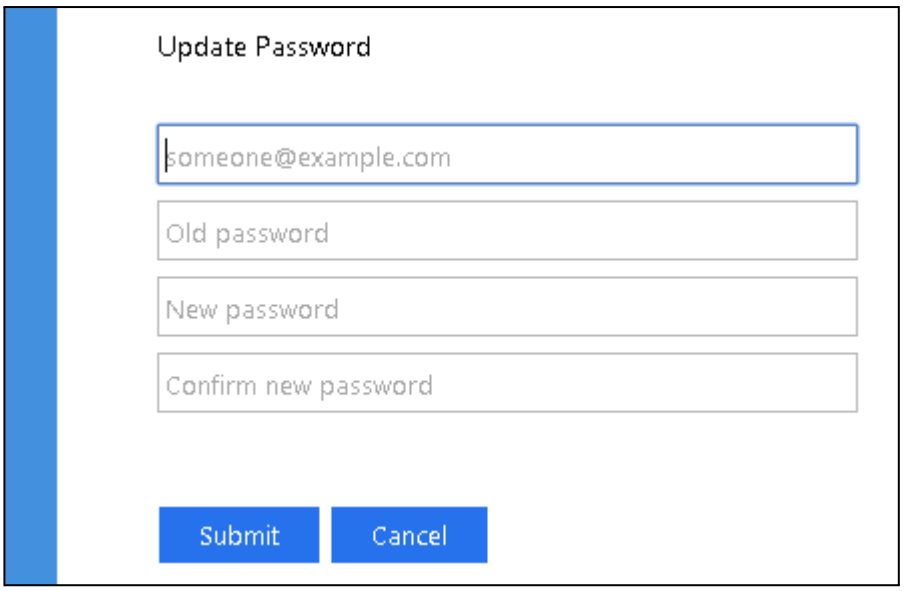


To enable password change, first you need to enable dedicated endpoint. Please go to the list of AD FS endpoints and scroll down. At the end of the endpoint list should be something like: "/adfs/portal/updatepassword/" see screen below. Enable that endpoint.

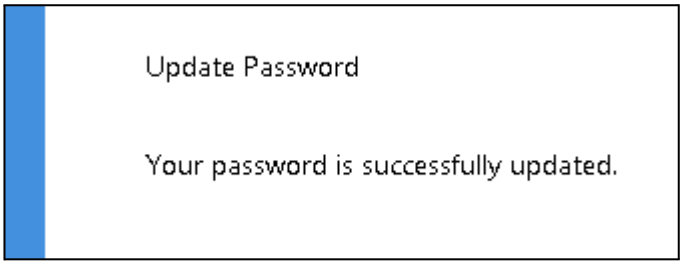


It's worth to mention that if AD FS service is on Windows 2012R2, password change page will be displayed only for users that are accessing from registered device. To bypass that limitation a special hotfix was released. See link below:
https://support.microsoft.com/en-us/help/3035025/hotfix-for-update-password-feature-so-that-users-are-not-required-to-use-registered-device-in-windows-server-2012-r2
When the AD FS is correctly configured, at the password change address user will get following form:

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

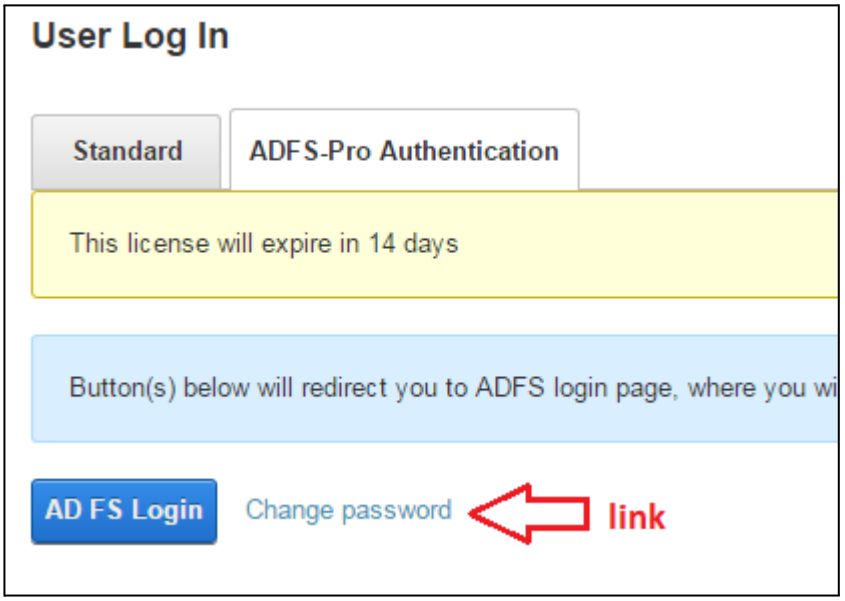


When password is successfully change following message is displayed:



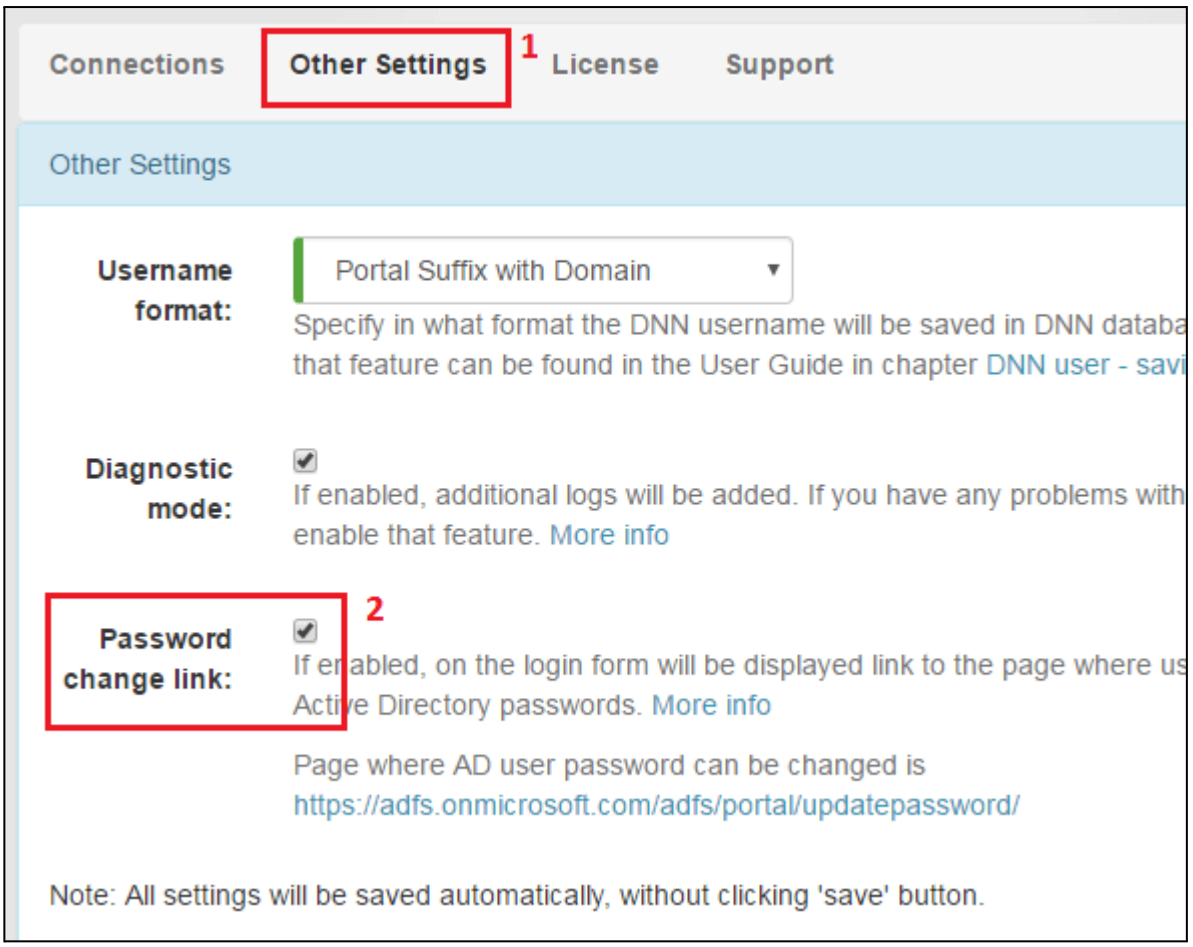
Module configuration

The "ADFS-Pro Authentication" can display link to the page where password will be changed. Please look at the following figure:



To enable/disable that link please edit provider, go to 'Other Options' tab and enable option 'Password change link'. See figure below.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



Vocabulary

Terminology used in ADFS

At this link can be found terminology that is related to the ADFS:
https://technet.microsoft.com/en-us/library/cc756089(v=ws.10).aspx

STS - Security Token Service

A Web service that issues security tokens. In ADFS, the Federation Service is an STS. A service itself can generate tokens or it can rely on a separate STS to issue a security token with its own trust statement. This forms the basis of trust brokering.

HRD

In query string will be added key HRD with value equal to “home realm”. Home realm URI will redirects the user to a particular IdP only and not provide an option to choose from
More info: http://www.cloudidentity.com/blog/2010/05/11/a-hidden-gem-the-wif-config-schema/
http://msdn.microsoft.com/en-us/library/microsoft.identitymodel.web.wsfederationauthenticationmodule_members.aspx

Claim

Think of a claim as a piece of identity information such as name, e-mail address, age, membership in the Sales role. The more claims your application receives, the more you’ll know about your user. You may be

wondering why these are called “claims,” rather than “attributes,” as is commonly used in describing enterprise directories. The reason has to do with the delivery method. In this model, your application doesn’t look up user attributes in a directory. Instead, the user delivers claims to your application, and your application examines them. Each claim is made by an issuer, and you trust the claim only as much as you trust the issuer. For example, you trust a claim made by your company’s domain controller more than you trust a claim made by the user herself. WIF represents claims with a Claim type, which has an Issuer property that allows you to find out who issued the claim [1].

SSO

Test Single SignOn Single SignOn is really nothing else but an application of the widely used “Remember Me” function. Say you have two web applications, SiteA and SiteB that share the same STS. You start the day by logging in to SiteA and do some work there. You’ll of course use the STS login page. The STS will establish a login session with the client. The browser will send the FedAuth cookie with all subsequent requests.
Then at some point you need to use SiteB. SiteB also needs authentication and redirects the user to the same STS. However, the STS will recognise the user’s FedAuth cookie and will issue another token for SiteB without having to log in again.
Therefore we get Single SignOn accross all applications that use the same STS.

Login params

- wa - the action to execute, which is wsignin1.0,
- wtrealm - the relying party that this token applies to, which is a-Expense,
- wctx - context data such as a return URL that will be propagated among the different parties, ADFS 2.0 uses guid when it generates wctx to identity provider (in case ADFS 2.0 is relying party for other STS). The context information is saved in a secure cookie that's identified by the guid that's later echoed back by identity provider
- wct - a time stamp,

Reference: http://msdn.microsoft.com/en-us/library/ff359114.aspx#sec1

MSISIPSelectionPersistent

MSISIPSelectionPersistent is the name of persistent cookie which is written to the file system on the client that shows who should be the identity provider (IDP) for this client. If the client does not already have this cookie set, and there are multiple IDPs to choose from, AD FS will prompt the user to select an IDP through a process called Home Realm Discovery (HRD).
The MSISIPSelectionPersistent cookie data is base64 encoded, so you can use your favorite base64 decoder to see the value of the identity provider. Fiddler has a base64 decoder built into its Decoders menu. Example of cookie data looks like: “http://sso.contoso.com/adfs/services/trust” (this is Federation Service Identifier. This is a URI, not a URL!)

MSISAuth

MSISAuth and MSISAuth1 are the encrypted cookies used to validate the SAML assertion produced for the client. These are what we call the "authentication cookies", and you will see these cookies ONLY when AD FS 2.0 is the IDP. Without these, the client will not experience SSO when AD FS 2.0 is the IDP.

MSISAuthenticated

[1] https://msdn.microsoft.com/en-us/library/hh873308(v=vs.110).aspx

MSISAuthenticated contains a base64-encoded timestamp value for when the client was authenticated. You will see this cookie set whether AD FS 2.0 is the IDP or not.

MSISSignout

MSISSignout is used to keep track of the IDP and all RPs visited for the SSO session. This cookie is utilized when a WS-Federation sign-out is invoked. You can see the contents of this cookie using a base64 decoder.

MSISLoopDetectionCookie

MSISLoopDetectionCookie is used by the AD FS 2.0 infinite loop detection mechanism to stop clients who have ended up in an infinite redirection loop to the Federation Server. For example, if an RP is having an issue where it cannot consume the SAML assertion from AD FS, the RP may continuously redirect the client to the AD FS 2.0 server. When the redirect loop hits a certain threshold, AD FS 2.0 uses this cookie to detect that threshold being met, and will throw an exception which lands the user on the AD FS 2.0 error page rather than leaving them in the loop. The cookie data is a timestamp that is base64 encoded.

ADFS Federation Metadata

Information about the services offered by an entity, usually the AD FS server. These information are served by following endpoint:
https://YourAdfsDomain/FederationMetadata/2007-06/FederationMetadata.xml
for example: https://adfs.server2012.org/FederationMetadata/2007-06/FederationMetadata.xml

Troubleshooting

Diagnostic Mode

The “Diagnostic Mode” displays info about the “ADFS-Pro Authentication” extension. These informations can help to diagnose issues that can occur like: config errors, failed login process, etc.

To enable “Diagnostic Mode” follow steps below:

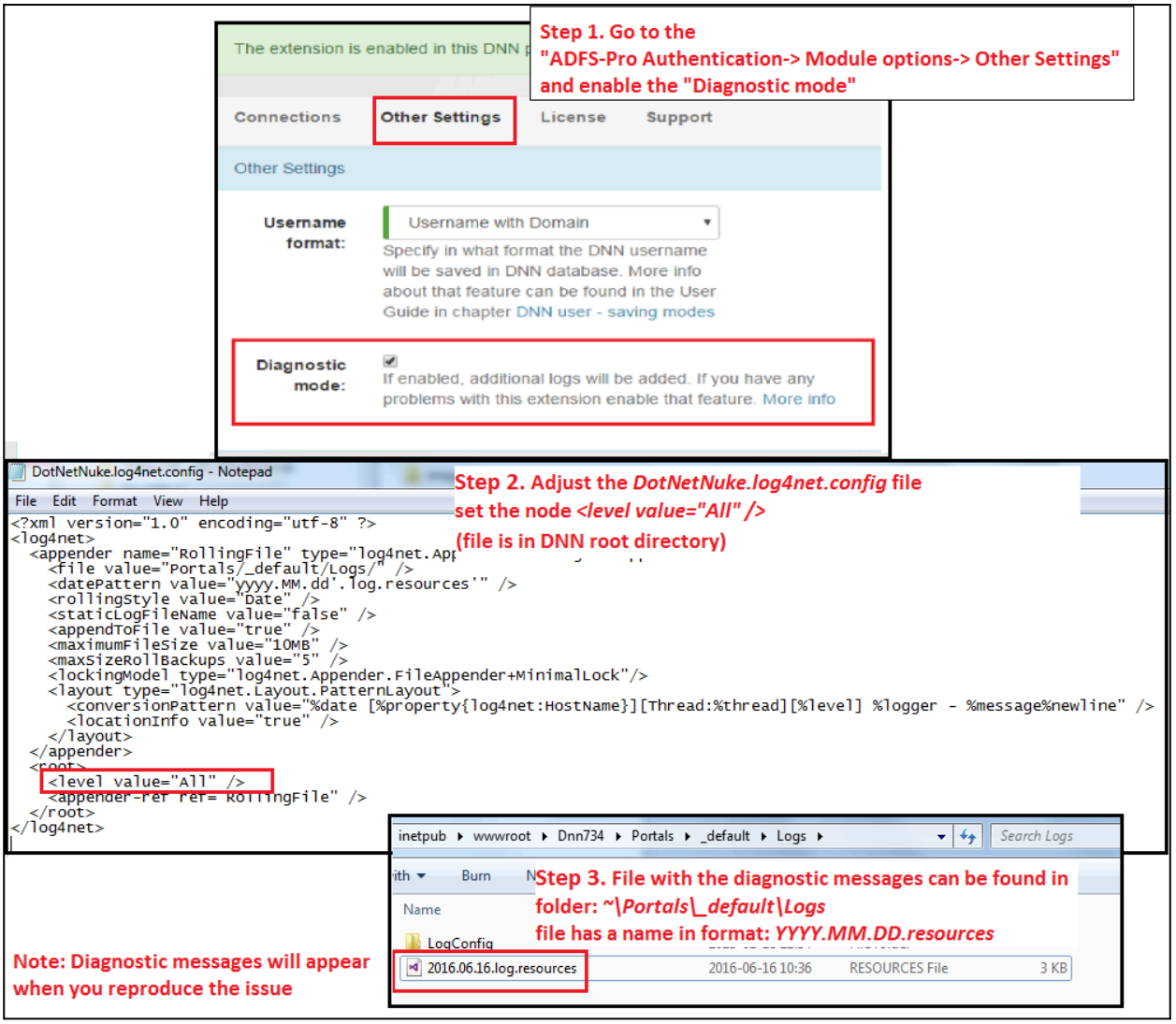
- edit log4net config file: DotNetNuke.log4net.config, usually can be found in root DNN Platform folder. Set the “level” to “ALL”

```
<root>
    <level value="ALL" />
    <appender-ref ref="RollingFile" />
</root>
```

- go to “ADFS-Pro Authentication-> Module options-> Other Settings” tab and enable attribute “Diagnostic Mode”, more info on figure below

When the “Diagnostic mode” is enabled, output logs can be found in folder `Portals\_default\Logs`. Log files are in the format `YYYY.MM.DD.logs.resources`.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



Java Script errors

The "AD-Pro Authentication" user interface is based on the JavaScript and html templates. If there are any issues it's worth to check are there any JavaScript errors. Please check following articles that are describing how to display these errors in your browser:

- JS errors in Chrome
- JS errors in FireFox
- JS errors in Internet Explorer

If you have any problems with "AD-Pro Authentication v3" module, please send error messages to support@glanton.com

Edit & Delete buttons doesn't work

When you can't update the module settings, and in web browser console you get the message like "Method Not Allowed" or 405 HTTP error code, please make sure that WebDAV is disabled. To disable WebDAV please add following lines to the web.config file:

- in the section "system.webServer-> modules" add following line:

```
<modules>
        <remove name="WebDAVModule"/> <!-- add this -->
        ...
</modules>
```

- in the section “system.webServer-> handlers” add following line:

```
<handlers>
        <remove name="WebDAV" />
        ...
</handlers>
```

You can read more about WebDAV here:
http://www.iis.net/learn/get-started/whats-new-in-iis-7/what39s-new-for-webdav-and-iis-7

Certificate is not in the trusted people store

*The X.509 certificate CN=*** is not in the trusted people store. The X.509 certificate CN=*** chain building failed. The certificate that was used has a trust chain that cannot be verified. Replace the certificate or change the certificateValidationMode. A certificate chain processed, but terminated in a root certificate which is not trusted by the trust provider.*

Follow these instructions:
http://msdn.microsoft.com/library/azure/jj192993(v=azure.10).aspx

Get info about actual AD FS

“Get-ADFSProperties” command executed in PowerShell will list info about current ADFS instance. More info: http://technet.microsoft.com/en-us/library/dn280950.aspx

To set a SPN

execute following command:
setspn -s host/{your_Federation_Service_name} {domain_name}\{service_account}
setspn -s http/W-Server12R2.cloudapp.net cloudapp.net\barry
(Remember to run your Command Prompt with elevated privileges or you will get an "Access Denied" message.)

No valid key mapping found for securityToken

If you get message like:

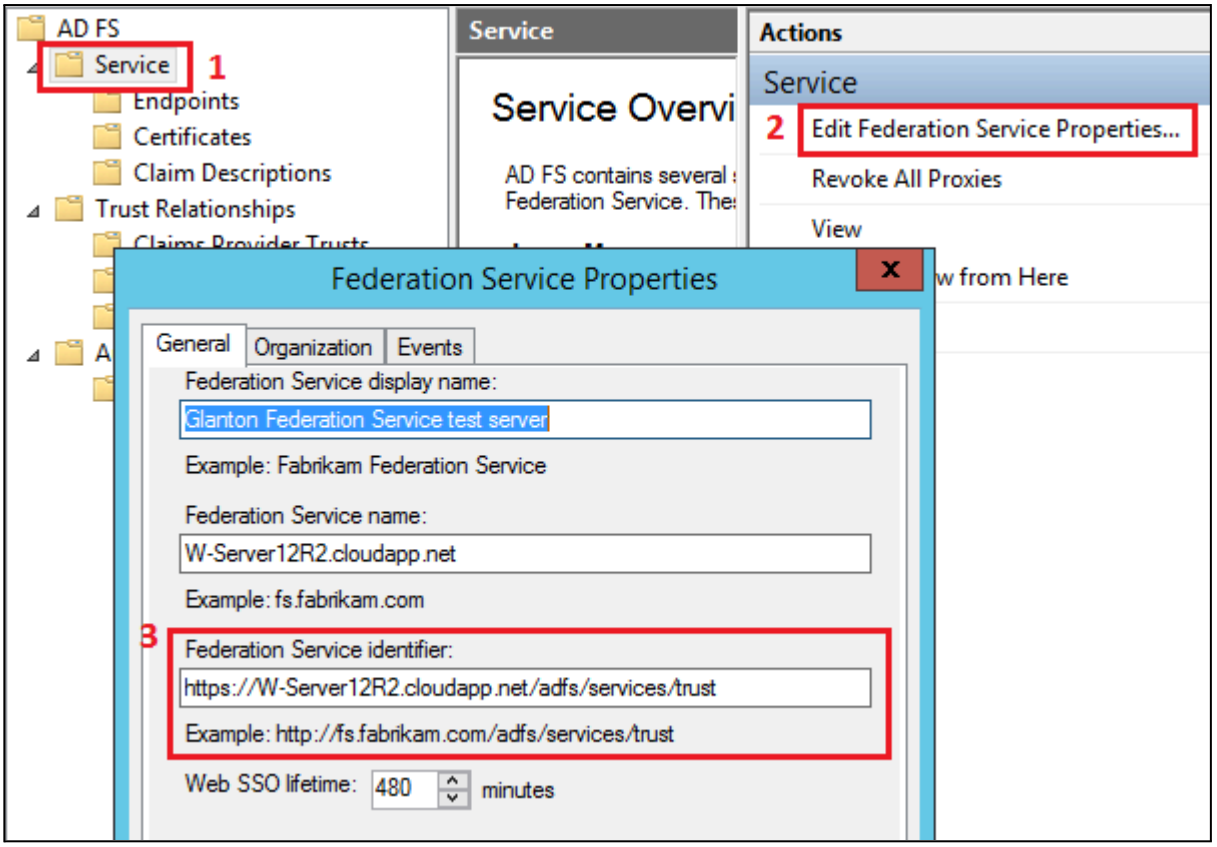
WIF10201: No valid key mapping found for securityToken: ‘System.IdentityModel.Tokens.X509SecurityToken' and issuer: 'https://W-Server12R2.cloudapp.net/adfs/services/trust'

Please make sure that in “ADFS-Pro Authentication” attributes:

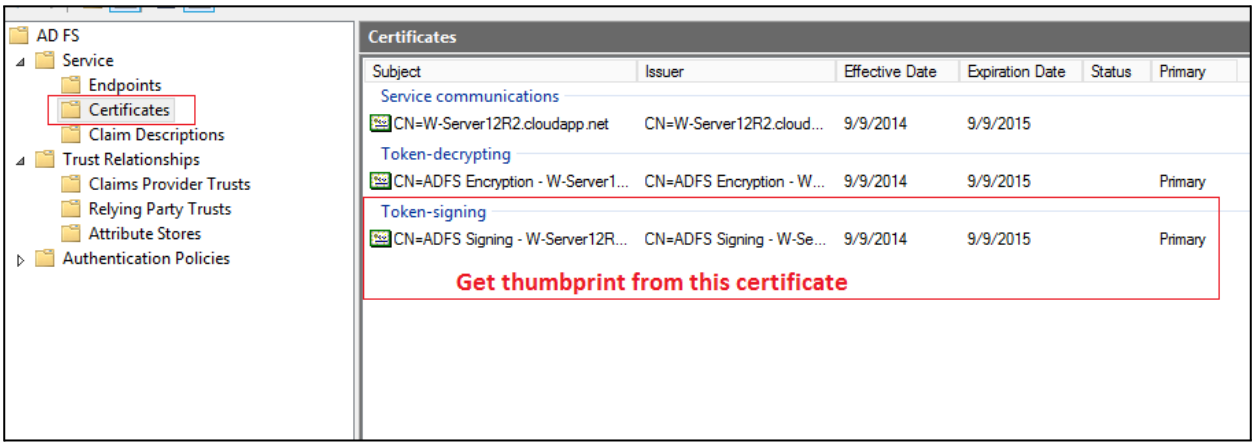
- Issuer Name Registry
- Valid Issuers

are equal to Federation Service Identifier in ADFS

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



and the certificate thumbprint is valid. To get the correct certificate, please follow steps from the picture below:

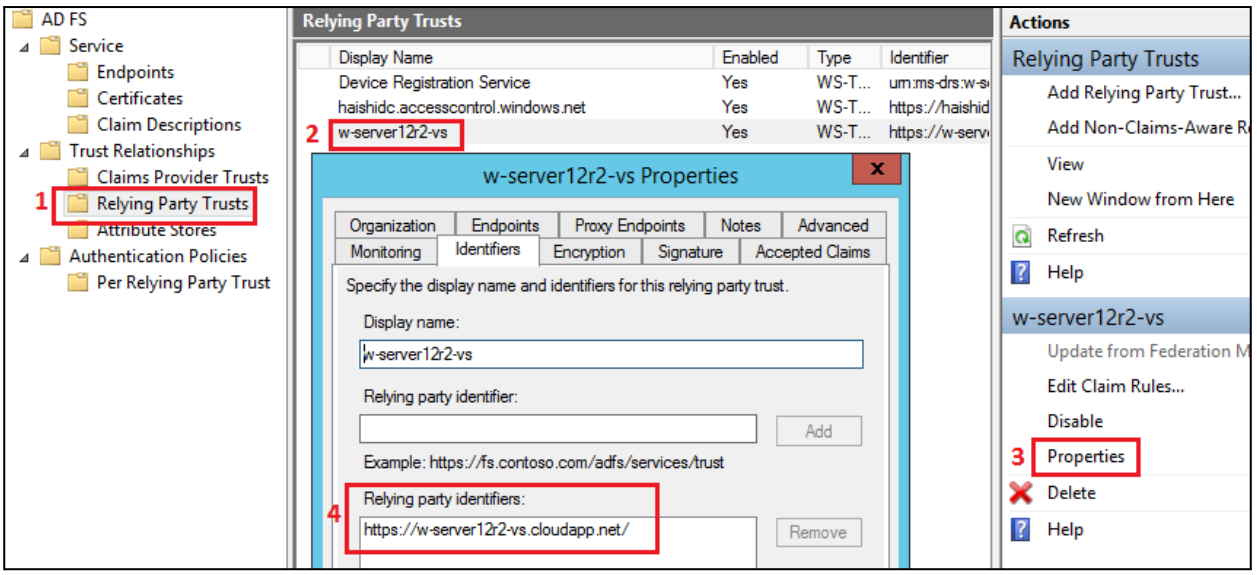


The requested relying party trust 'https://...' is unspecified

If in Event Viewer on ADFS server, you get message like:
Microsoft.IdentityServer.Web.InvalidScopeException: MSIS7007: The requested relying party trust 'https://w-server12r2-vs.cloudapp.net/' is unspecified or unsupported. If a relying party trust was specified, it is possible that you do not have permission to access the trust relying party. Contact your administrator for details.

Please make sure that Realm specified in DNN in “ADFS-Pro Authentication” provider is equal to Relying Party Identifier in ADFS (screen below).

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



The Audience Restriction Condition was not valid

If you get the message like:

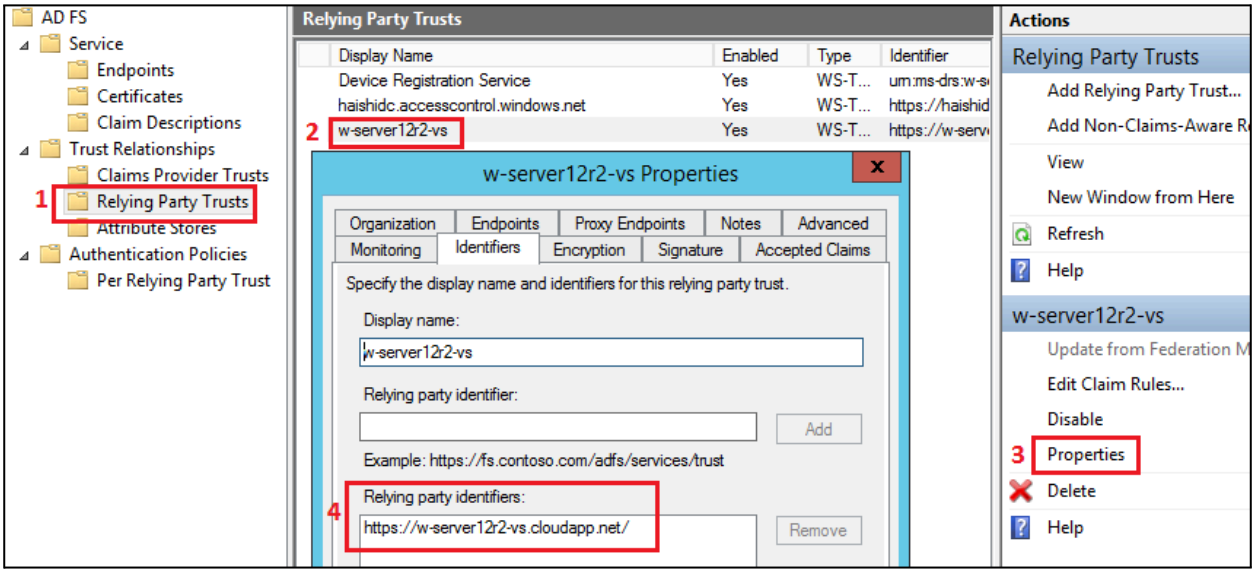
ID1038: The AudienceRestrictionCondition was not valid because the specified Audience is not present in AudienceUris.Audience: 'https://MyDnn.cloudapp.net/'

Please make sure that following values are equal:

Audience Uris - in WS Provider,

Realm - in WS Provider,

Relying Party Identifier - in ADFS (screen below);



URL scheme is not https

If you get the message like:

ID1059: Cannot authenticate the user because the URL scheme is not https and requireSsl is set to true in the configuration, therefore the authentication cookie will not be sent. Change the URL scheme to https or set requireSsl to false on the cookieHandler element in configuration.

Please make sure that the DNN is using https, or set “require SSL attribute to false

<cookieHandler requireSsl="false" />

Issuer of the security token was not recognized by the IssuerNameRegistry

If you get the message like:
ID4175: The issuer of the security token was not recognized by the IssuerNameRegistry. To accept security tokens from this issuer, configure the IssuerNameRegistry to return a valid name for this issuer.

Please make sure that the following attributes are correct:

- certificate thumbprint,
- Issuer Name Registry;

Could not load the identity configuration

If you get the message like:

ID7027: Could not load the identity configuration because no <system.identityModel> configuration section was found.

Please make sure that web.config file has this lines:

<section name="system.identityModel" type="System.IdentityModel.Configuration.SystemIdentityModelSection, System.IdentityModel, Version=4.0.0.0, Culture=neutral, PublicKeyToken=B77A5C561934E089" />
<section name="system.identityModel.services" type="System.IdentityModel.Services.Configuration.SystemIdentityModelServicesSection, System.IdentityModel.Services, Version=4.0.0.0, Culture=neutral, PublicKeyToken=B77A5C561934E089" />

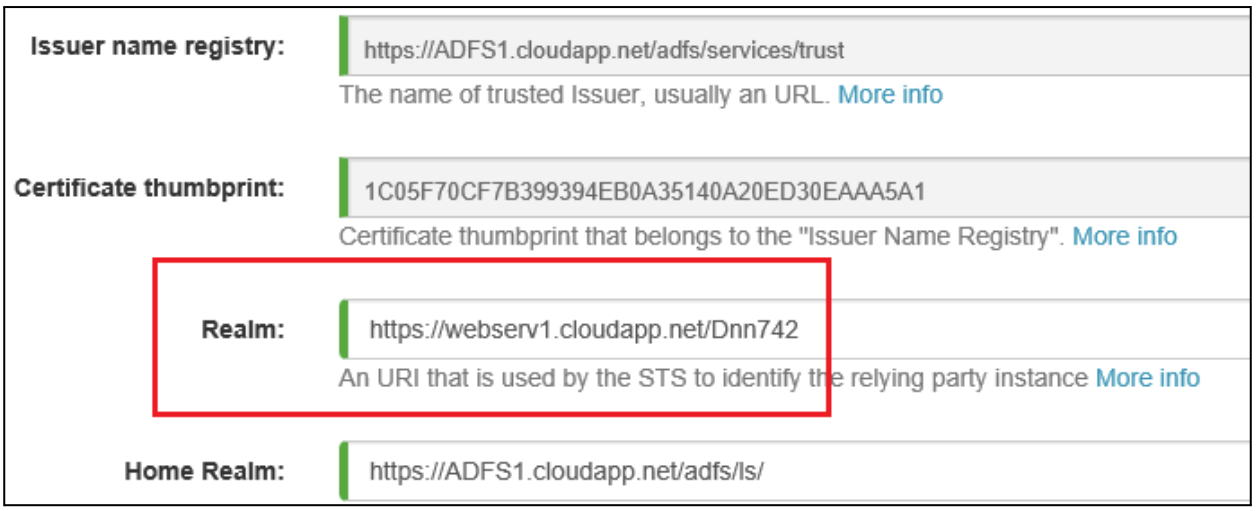
STS address is not configured

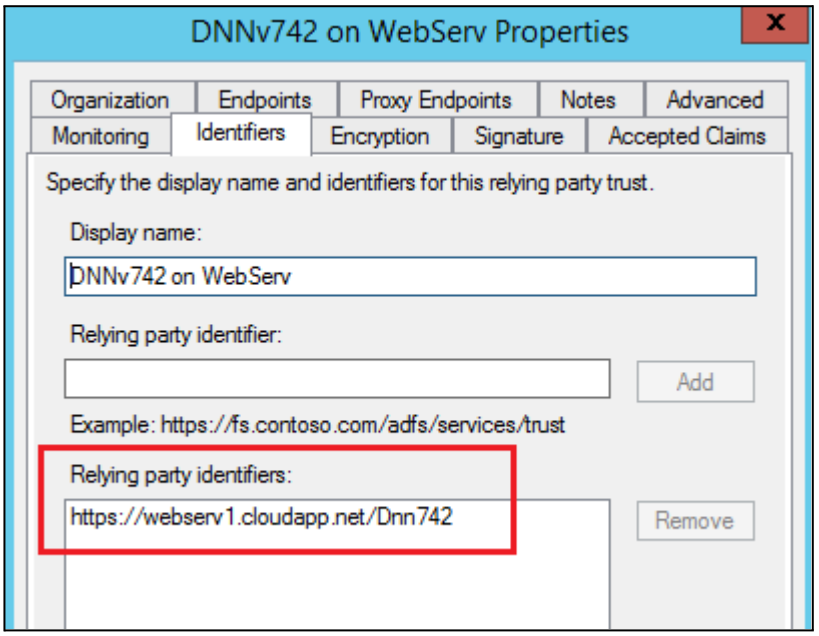
If you get the message like:

ID1058: A valid STS address is not configured on the WSFederationAuthenticationModule for creating WS-Federation passive protocol SignOut requests. Set the Issuer property on the module to a valid STS address.

Please make sure that: Realm field in the “ADFS-Authentication” module and the “Relying party identifier” in the ADFs config is exactly the same.

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html





A SignInResponse message may only redirect within the current web application

If you get message like:
ID3206: A SignInResponse message may only redirect within the current web application: 'https://MyDnn.cloudapp.net' is not allowed.
A return url address must have "/" at the end. In the logs should be info about your current url.

There are no registered protocol handlers on path /adfs/ls/

If in AD Event Viewer you get the message like:
Microsoft.IdentityServer.RequestFailedException:

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

MSIS7065: There are no registered protocol handlers on path /adfs/ls/ to process the incoming request.
at Microsoft.IdentityServer.Web.PassiveProtocolListener.OnGetContext(WrappedHttpListenerContext context)
Resolution:

1. Ensure that SPN are correct.
2. Ensure that the certificate is not expired.

WebForms UnobtrusiveValidationMode

If you get the exception like:
System.InvalidOperationException: WebForms UnobtrusiveValidationMode requires a ScriptResourceMapping for 'jquery'. Please add a ScriptResourceMapping named jquery(case-sensitive).
Please make sure that in web.config file in "appSettings" section you have following entry:

```
<appSettings>
...
<add key="ValidationSettings:UnobtrusiveValidationMode" value="None" />
...
</appSettings>
```

If this key value is set to "None" [default], the ASP.NET application will use the pre-4.5 behavior (JavaScript inline in the pages) for client-side validation logic. More info:
https://msdn.microsoft.com/en-us/library/hh975440.aspx

Changes in web.config

The "ADFS-Pro Authentication" automatically creates following changes in web.config file:

1. Under section <configuration><configSections> following lines will be added:

```
<section name="system.identityModel"
type="System.IdentityModel.Configuration.SystemIdentityModelSection, System.IdentityModel,
Version=4.0.0.0, Culture=neutral, PublicKeyToken=B77A5C561934E089" />
<section name="system.identityModel.services"
type="System.IdentityModel.Services.Configuration.SystemIdentityModelServicesSection,
System.IdentityModel.Services, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=B77A5C561934E089" />
```

```
    <section name="sitemap" requirePermission="false" type="DotNetNuke.Framework.Providers.ProviderConfigurationHandler, DotNetNuk
    <section name="cryptography" requirePermission="false" type="DotNetNuke.Framework.Providers.ProviderConfigurationHandler, DotN
  </sectionGroup>

  <section name="system.identityModel"
           type="System.IdentityModel.Configuration.SystemIdentityModelSection, System.IdentityModel,
                 Version=4.0.0.0, Culture=neutral, PublicKeyToken=B77A5C561934E089" />
  <section name="system.identityModel.services"
           type="System.IdentityModel.Services.Configuration.SystemIdentityModelServicesSection, System.IdentityModel.Services,
                 Version=4.0.0.0, Culture=neutral, PublicKeyToken=B77A5C561934E089" />

  <sectionGroup name="system.web.webPages.razor" type="System.Web.WebPages.Razor.Configuration.RazorWebSectionGroup, System.Web.We
    <section name="pages" type="System.Web.WebPages.Razor.Configuration.RazorPagesSection, System.Web.WebPages.Razor" requirePermi
  </sectionGroup>
  <section name="clientDependency" type="ClientDependency.Core.Config.ClientDependencySection, ClientDependency.Core" requirePermi
</configSections>
```

2. Under section <configuration><system.webServer><modules> following lines will be added:

```
<add name="CustomWSFederationAuthenticationModule"
type="GS.FederationProvider.CustomWSFederationAuthenticationModule, GS.FederationProvider"
preCondition="managedHandler" />

<add name="SessionAuthenticationModule"
type="System.IdentityModel.Services.SessionAuthenticationModule,
```

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

```
System.IdentityModel.Services, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089" preCondition="managedHandler" />
```

```
<add name="Detector" type="FiftyOne.Foundation.Mobile.Detection.DetectorModule, FiftyOne.Foundation" preCondition="managedHandler" />
<add name="RadUploadModule" type="Telerik.Web.UI.RadUploadHttpModule, Telerik.Web.UI" preCondition="managedHandler" />

<add name="CustomWSFederationAuthenticationModule"
     type="GS.FederationProvider.CustomWSFederationAuthenticationModule, GS.FederationProvider"
     preCondition="managedHandler" />
<add name="SessionAuthenticationModule"
     type="System.IdentityModel.Services.SessionAuthenticationModule,
           System.IdentityModel.Services, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"
     preCondition="managedHandler" />
<add name="SessionAuthenticationModule"
     type="System.IdentityModel.Services.SessionAuthenticationModule,
           System.IdentityModel.Services, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"
     preCondition="managedHandler" />

</modules>
<handlers>
```

Note: old versions of "ADFS-Pro Authentication" module creates following lines in web.config, in current module version these lines are invalid. These lines should be automatically removed if you installing v1.5.7+ . If lines still exist remove them manually.

```
<add name="AuthModule" type="GS.FederationProvider.WSFederationModule, GS.FederationProvider"
/>

<add name="WSFederationAuthenticationModule"
type="GS.FederationProvider.WSFederationAuthenticationModule, GS.FederationProvider"
preCondition="managedHandler" />
```

WebAPI request are not supported

Too many redirects issue occur

Fix -> custom Dotnetnuke.Modules.dll

CryptographicException occurred - cookie encrypt

If you get the message like:

ID1074: A CryptographicException occurred when attempting to encrypt the cookie using the ProtectedData API (see inner exception for details). If you are using IIS 7.5, this could be due to the loadUserProfile setting on the Application Pool being set to false.

The data protection operation was unsuccessful. This may have been caused by not having the user profile loaded for the current thread's user context, which may be the case when the thread is impersonating.

Please make sure that the following code is in your web.config file:

```
<system.identityModel>
   <identityConfiguration>
     <securityTokenHandlers>
        <remove    type="System.IdentityModel.Tokens.SessionSecurityTokenHandler,    System.IdentityModel,
Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" />
     <add type="System.IdentityModel.Services.Tokens.MachineKeySessionSecurityTokenHandler,
System.IdentityModel.Services, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" />
      </securityTokenHandlers>
   </identityConfiguration>
</system.identityModel>
```

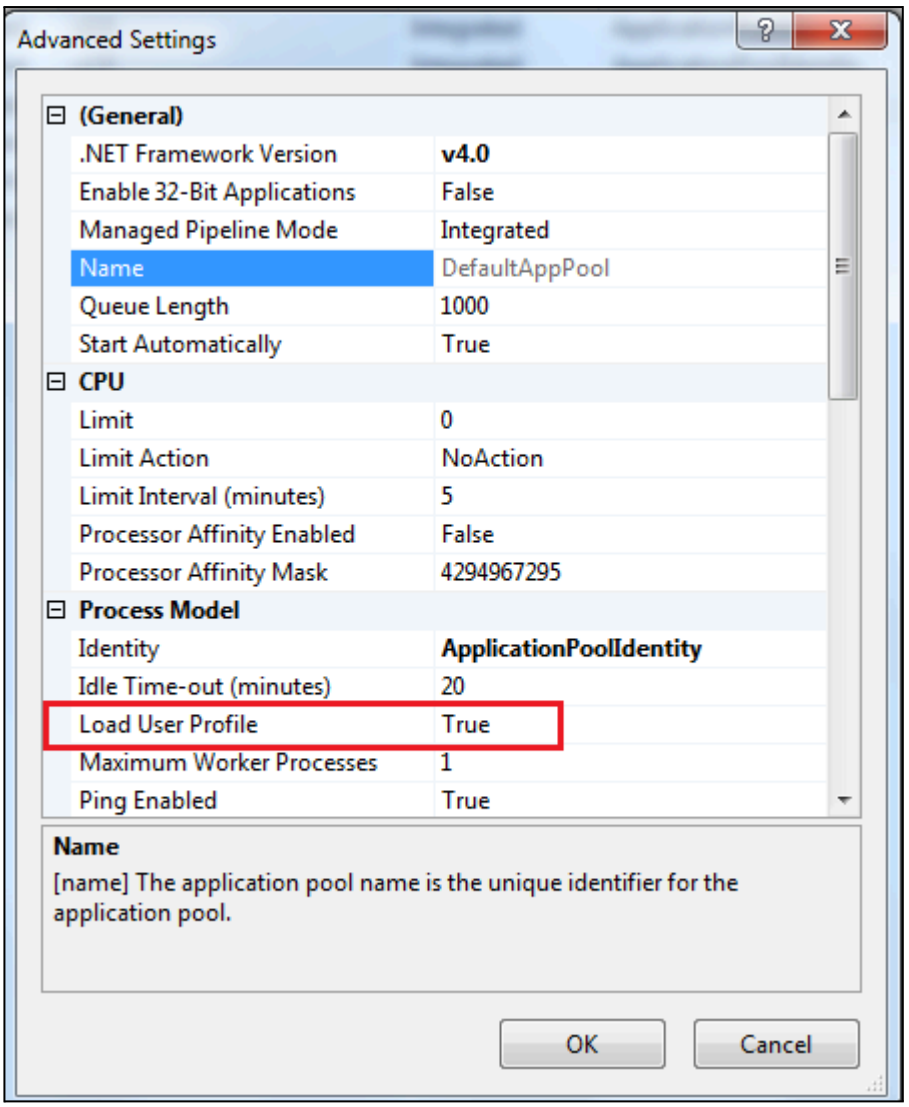
More info can be found here:

http://www.cloudidentity.com/blog/2013/01/28/running-wif-based-apps-in-windows-azure-web-sites-4/

If DNN is hosted on a web farm or behind the loadbalancer read this chapter: Session token encryption

Another important note: make sure that AppPool has enabled "Load User Profile", see image below:

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html



CryptographicException occurred - cookie decrypt

If you get an error message like:

System.InvalidOperationException: ID1073: A CryptographicException occurred when attempting to decrypt the cookie using the ProtectedData API (see inner exception for details). If you are using IIS 7.5, this could be due to the loadUserProfile setting on the Application Pool being set to false.

We need to implement a new solution of encryption and decryption session tokens.

This could happen when DNN is hosted on the server that is behind the load balancer. In this case server is deployed on multiple machines for example on a web farm.

When user continue to browse the site (after login) the load balancer redirects some request to execute on machine-1 some on machine-2. If machine-1 receives the first request after login then WIF uses machine-1's current user key to encrypt the cookie. When the next request goes to machine-2, WIF in machine-2 will fail to decrypt the cookie using machine-2's current user key and you will get the error like above.

To fix that issue a certificate need to be used to decrypt/encrypt the session tokens. Add following entry in web.config file to use certificate:

```
<configuration>
    <system.identityModel.services>
```

This document is deprecated. New user guide is here:
http://doc.glanton.com/ADFS-Pro-Authentication/index.html

```
        <federationConfiguration>
            <serviceCertificate>
              <certificateReference x509FindType="FindByThumbprint" findValue="<cerrt-thumbnail>"
                                    storeLocation="LocalMachine" storeName="My"/>
            </serviceCertificate>
        </federationConfiguration>
    </system.identityModel.services>
</configuration>
```

Reference: http://anuchandy.blogspot.com.es/2014/06/acs-wif-asp-net-and-microsoft-azure.html

References

ADFS-Pro Authentication - DNN plugin that allows you connect DNN to AD FS
DNN&ADFS - User Guide that describes implementation details

- https://msdn.microsoft.com/en-us/library/hh568665(v=vs.110).aspx
- http://dotnetcodr.com/2013/03/11/claims-based-authentication-in-net4-5-mvc4-with-c-external-authentication-with-ws-federation-part-2-testing-a-real-sts/
- Whitepaper: Understanding WS-Federation

Benefits of ADFs in the Azure Virtual machine:

https://docs.microsoft.com/en-us/azure/active-directory/connect/active-directory-aadconnect-azure-adfs