Den Europæiske Union ved Den Europæiske Fond for Udvikling af Landdistrikter og Ministeriet for Fødevarer, Landbrug og Fiskeri har deltaget i finansieringen af projektet.

VIDENCENTRET FOR LANDBRUG

Configuring WCF Service application to use AD FS 2.0

A step-by-step guide

Indhold

1 Dokumentinformation

1.1 Status

Status	Beskrivelse
Udkast	Dokumentet er under udarbejdelse.

1.2 Versionshistorik

Dato	Version	Initialer	Beskrivelse
07-01-2013	1.0	TMC	Første version af dokumentet.

2 Configure WCF application to use AD FS 2.0 - A step-by-step guide

This guide assumes that the following components are installed and configured on the development server:

1. IIS version 7.0 or later
2. Visual Studio 2010 or later
3. Windows Identity Foundation 3.5 and Windows Identity Foundation SDK 4.0 (elaborated in step 2)
4. NuGet Package Manager

If the WCF application is using the deprecated DLI-SSO federation service as a claims provider, this configuration must be removed from the configuration file and any reference to DLI-SSO assemblies must be removed from the C# project before the application can be AD FS 2.0 enabled.

2.1 Step 1 – Create IIS web site and configure it to use SSL

Note: If a web site with host name “localhost.vfltest.dk” and configured with SSL certificate “*.vfltest.dk” already exists, this step can be skipped.

Prerequisites:

*.vfltest.dk SSL certificate file (and password)

To ease installation of SSL certificate, creation of IIS application pool and web site, a PowerShell script is available (TFS: $/DLBRLogin/DLBRLogin/trunk/Tools/Scripts/CreateWebsite.ps1).

The script does the following:

1. Installs SSL certificate.
2. Creates IIS Application Pool named “localhost.vfltest.dk”, supporting .NET Framework Version 4.0 running in Integrated Pipeline Mode with identity “NetworkService”.
3. Creates a web site named “localhost.vfltest.dk” supporting HTTPS bindings.
4. Attaches certificate “*.vfltest.dk” to port 0.0.0.0:443 (HTTPS binding). If another certificate is already bound to this port, the definition will be overridden.

Note: It is important that PowerShell is executed with administrator privileges. This can be accomplished by using the “RunAs” command, where the selected user is member of “Administrators” group on the machine, e.g. runas /user:xyz powershell.

To execute the script, go to the Windows start menu and type PowerShell and select any version of Windows PowerShell:

1. If this is the first time a PowerShell script is executed on the computer, an error will occur saying that “File C:\ CreateWebsite.ps1 cannot be loaded because the execution of scripts is disabled on this system…”. The following PowerShell command must be executed in the PowerShell window: “Set-ExecutionPolicy Unrestricted”.
2. Execute the script by typing the full path to “CreateWebsite.ps1”.
3. If the SSL certificate is already installed on the computer, the script outputs “SSL certifikat 'CN=*.vfltest.dk, OU=Domain Control Validated, C=DK' er allerede installeret i store LocalMachine\My (Local Computer – Personal). If not, you will be prompted for the path to the PFX file and the password to the private key.
4. Next you will be prompted for a name for the web site. Default value is “localhost.vfltest.dk”. Note that “.vfltest.dk” part of the domain name is required to comply with the SSL certificate subject name “*.vfltest.dk”.
5. If no errors occurred during script execution, the web site is now ready and running.

2.2 Step 2 - Installation of Windows Identity Foundation (WIF) 3.5 and WIF SDK 4.0

Before development of a claims aware WCF application, Windows Identity Foundation 3.5 (WIF) and the WIF SDK 4.0 must be installed.

WIF 3.5 is part of .NET 3.5 and can be downloaded from http://www.microsoft.com/en-us/download/details.aspx?id=17331.

Follow the instructions on the download page. WIF 3.5 must be installed prior to WIF SDK 4.0.

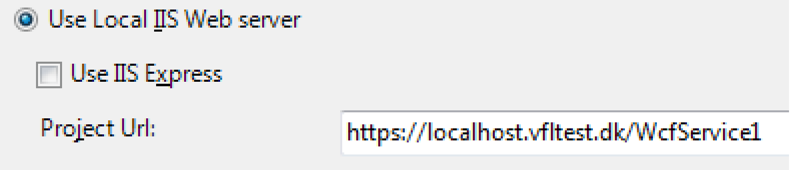
WIF SDK 4.0 can be downloaded from http://www.microsoft.com/en-us/download/details.aspx?id=4451. Choose the SDK for .NET 4.0. Note that side by side installation of the WIF SDK 3.5 and WIF SDK 4.0 is not recommended.

2.3 Step 3 – Create a claims aware WCF application

First a discussion about which version of Visual Studio to use. Visual Studio 2010 and Visual Studio 2012 both has support for .NET 4.0 applications, but only Visual Studio 2010 has built-in support for integrating a WCF application with WIF 3.5. This integration is surfaced by the “Add STS Reference…” command, that is available when you in Solution Explorer right-click the web application project file. If you prefer to use Visual Studio 2012, you have to use the external WIF SDK 4.0 tool “FedUtil” (in Visual Studio 2010 “FedUtil” is a built-in extension that is invoked by “Add STS Reference…” command). FedUtil can be found in the path C:\Program Files (x86)\Windows Identity Foundation SDK\v4.0\FedUtil.exe.

This guide assume that the project configuration file (web.config) is empty.

1. In Visual Studio, create a new empty WCF Service application.
2. In Solution Explorer, right-click the project and select “Properties”.
3. In the left pane click “Web”.
4. In the “Servers” section select “Use Local IIS Web Server. Make sure that “Use IIS Express is not selected, we want to use the IIS web site we created in the previous step. It is important that the domain part of the project url is “localhost.vfltest.dk” and the URI scheme is https. Choose a relevant application name instead of “WcfService1”. Click “Create Virtual Directory”. This will create a new IIS web application hosted in the “localhost.vfltest.dk” web site.

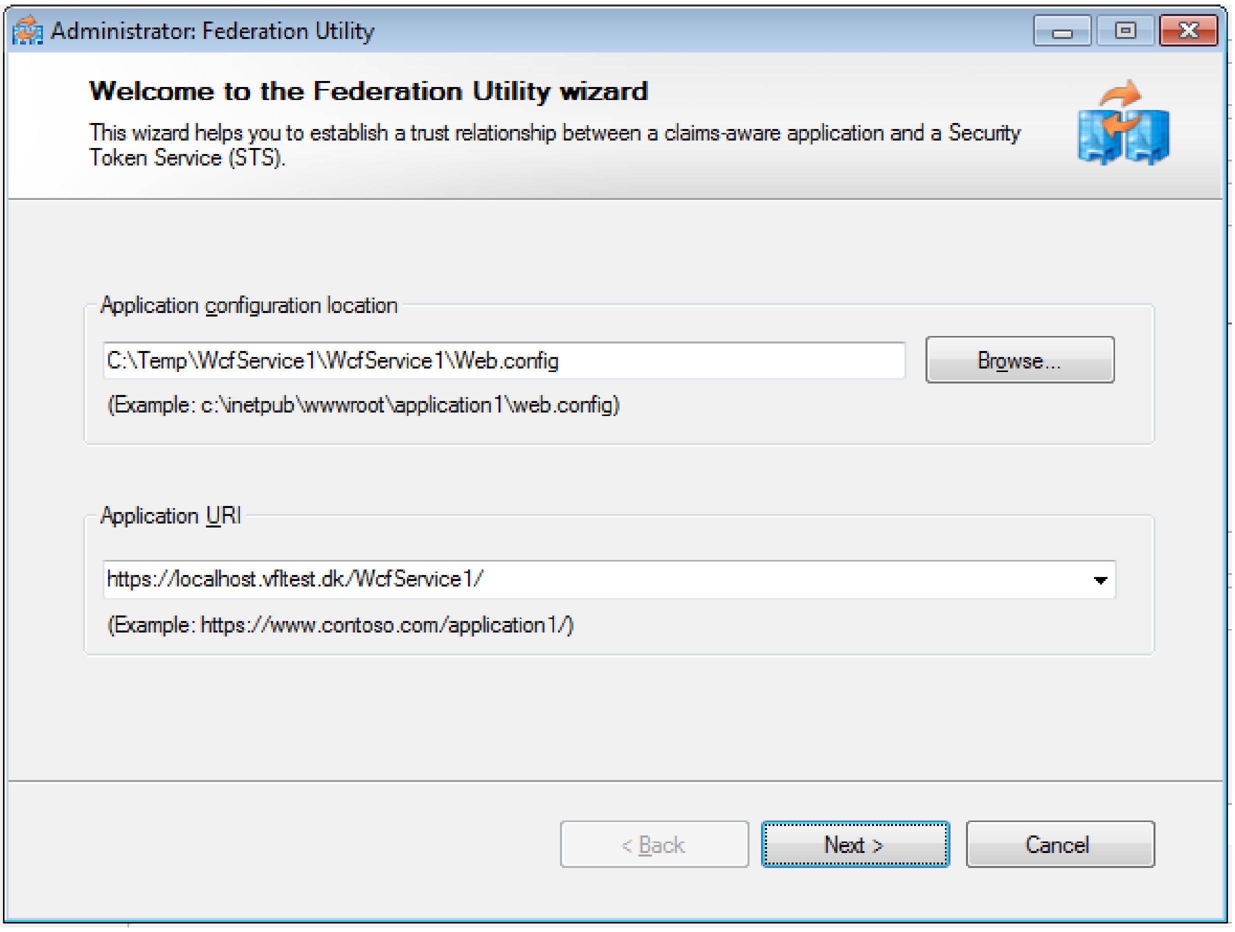


● Use Local IIS Web server

□ Use IIS Express

Project Url: https://localhost.vfltest.dk/WcfService1

5. If Visual Studio 2010, in Solution Explorer, right-click the project file and choose menu item "Add STS Reference…". If Visual Studio 2012, execute the external application "". Add the path to the web application configuration file (web.config) and the application URI. Note the trailing slash (/) in "Application URI".



Welcome to the Federation Utility wizard

This wizard helps you to establish a trust relationship between a claims-aware application and a Security Token Service (STS).

Application configuration location

C:\Temp\WcfService1\WcfService1\Web.config

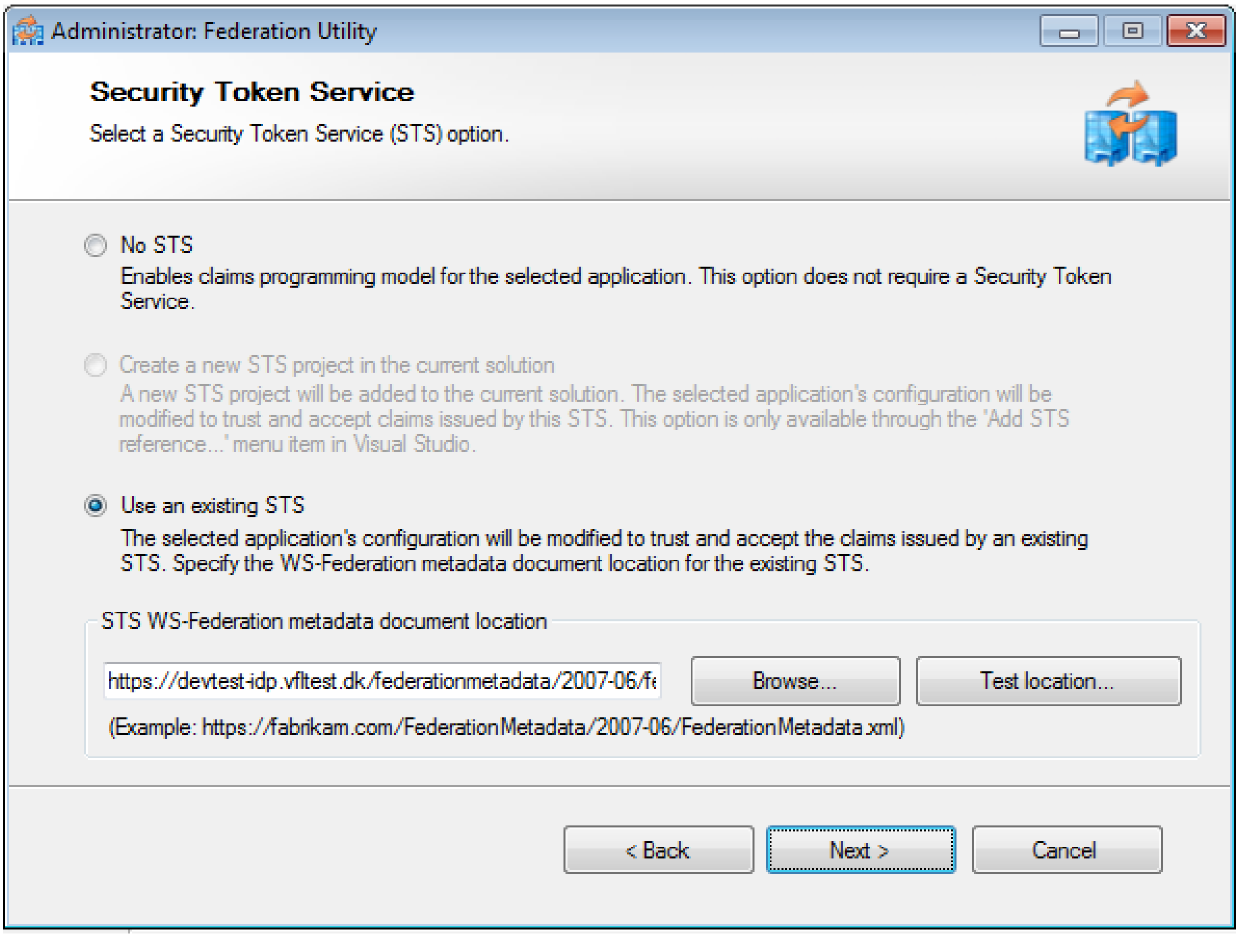
(Example: c:\inetpub\wwwroot\application1\web.config)

Application URI

https://localhost.vfltest.dk/WcfService1/

(Example: https://www.contoso.com/application1/)

6. Choose “Use an existing STS” and type “https://dev-idp.vfltest.dk/federationmetadata/2007-06/federationmetadata.xml“ or “https://devtest-idp.vfltest.dk/federationmetadata/2007-06/federationmetadata.xml“ as STS WS-Federation metadata document location, depending on whether the application must federate with DEV or DEVTEST identity provider.



Administrator: Federation Utility

Security Token Service

Select a Security Token Service (STS) option.

○ No STS
Enables claims programming model for the selected application. This option does not require a Security Token Service.

○ Create a new STS project in the current solution
A new STS project will be added to the current solution. The selected application's configuration will be modified to trust and accept claims issued by this STS. This option is only available through the 'Add STS reference...' menu item in Visual Studio.

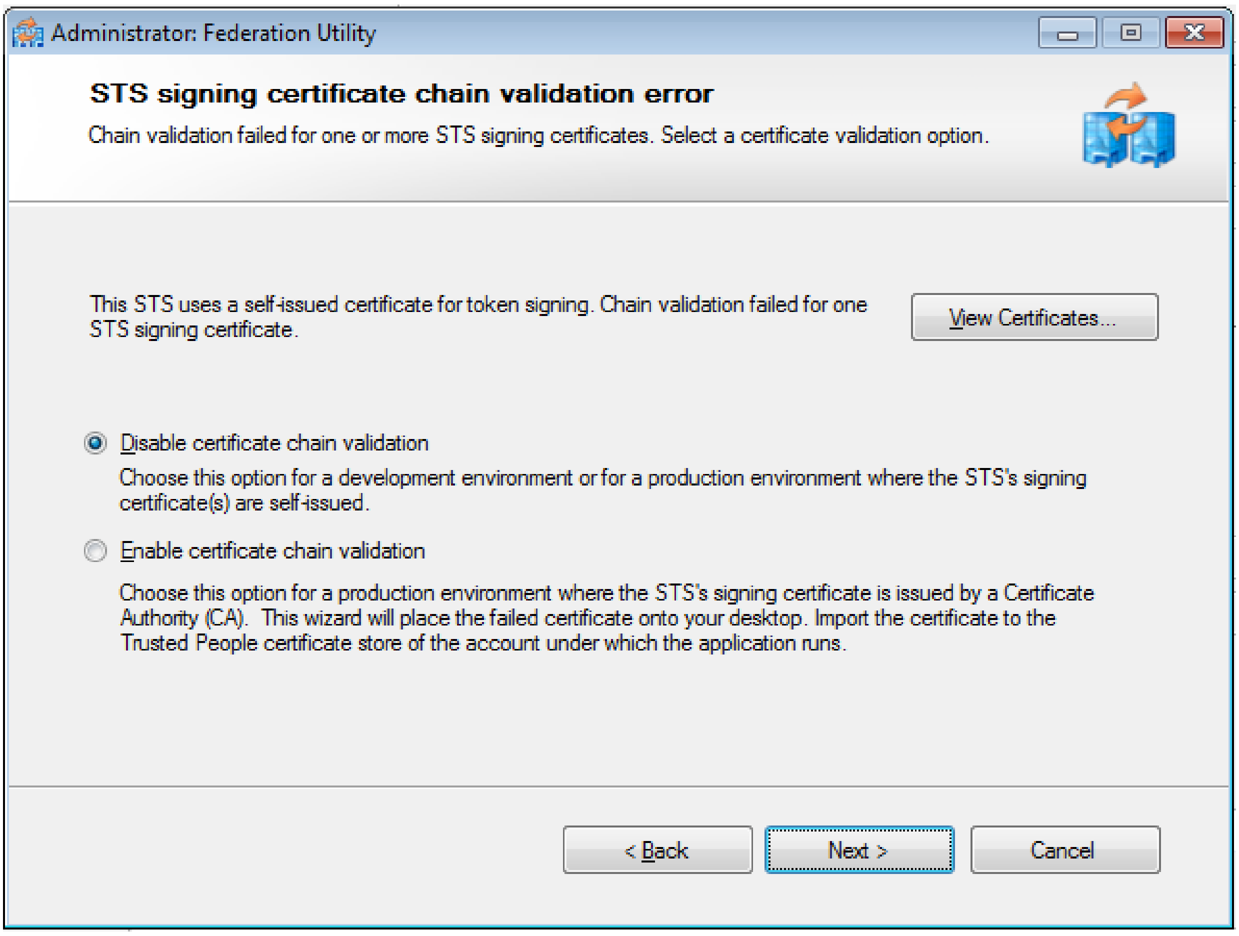
● Use an existing STS
The selected application's configuration will be modified to trust and accept the claims issued by an existing STS. Specify the WS-Federation metadata document location for the existing STS.

STS WS-Federation metadata document location

https://devtest-idp.vfltest.dk/federationmetadata/2007-06/fe

(Example: https://fabrikam.com/FederationMetadata/2007-06/FederationMetadata.xml)

7. Choose “Disable certificate chain validation”.



STS signing certificate chain validation error

Chain validation failed for one or more STS signing certificates. Select a certificate validation option.

This STS uses a self-issued certificate for token signing. Chain validation failed for one STS signing certificate.

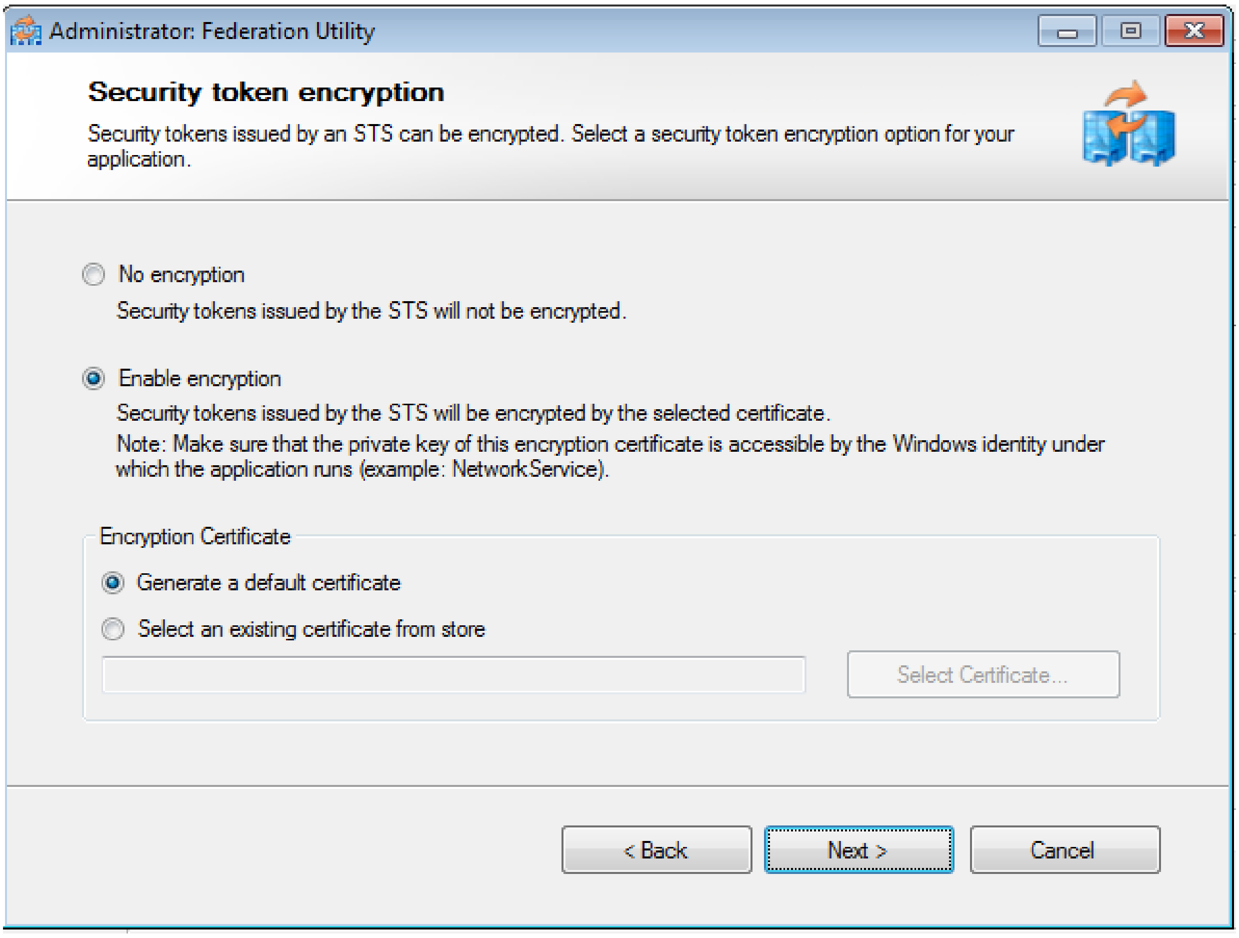
● Disable certificate chain validation

Choose this option for a development environment or for a production environment where the STS's signing certificate(s) are self-issued.

○ Enable certificate chain validation

Choose this option for a production environment where the STS's signing certificate is issued by a Certificate Authority (CA). This wizard will place the failed certificate onto your desktop. Import the certificate to the Trusted People certificate store of the account under which the application runs.

8. Choose "Enable encryption" even though we do not want to use encryption. The wizard does not allow WCF applications not to have encryption. We will remove encryption later.



Administrator: Federation Utility

Security token encryption

Security tokens issued by an STS can be encrypted. Select a security token encryption option for your application.

○ No encryption

Security tokens issued by the STS will not be encrypted.

● Enable encryption

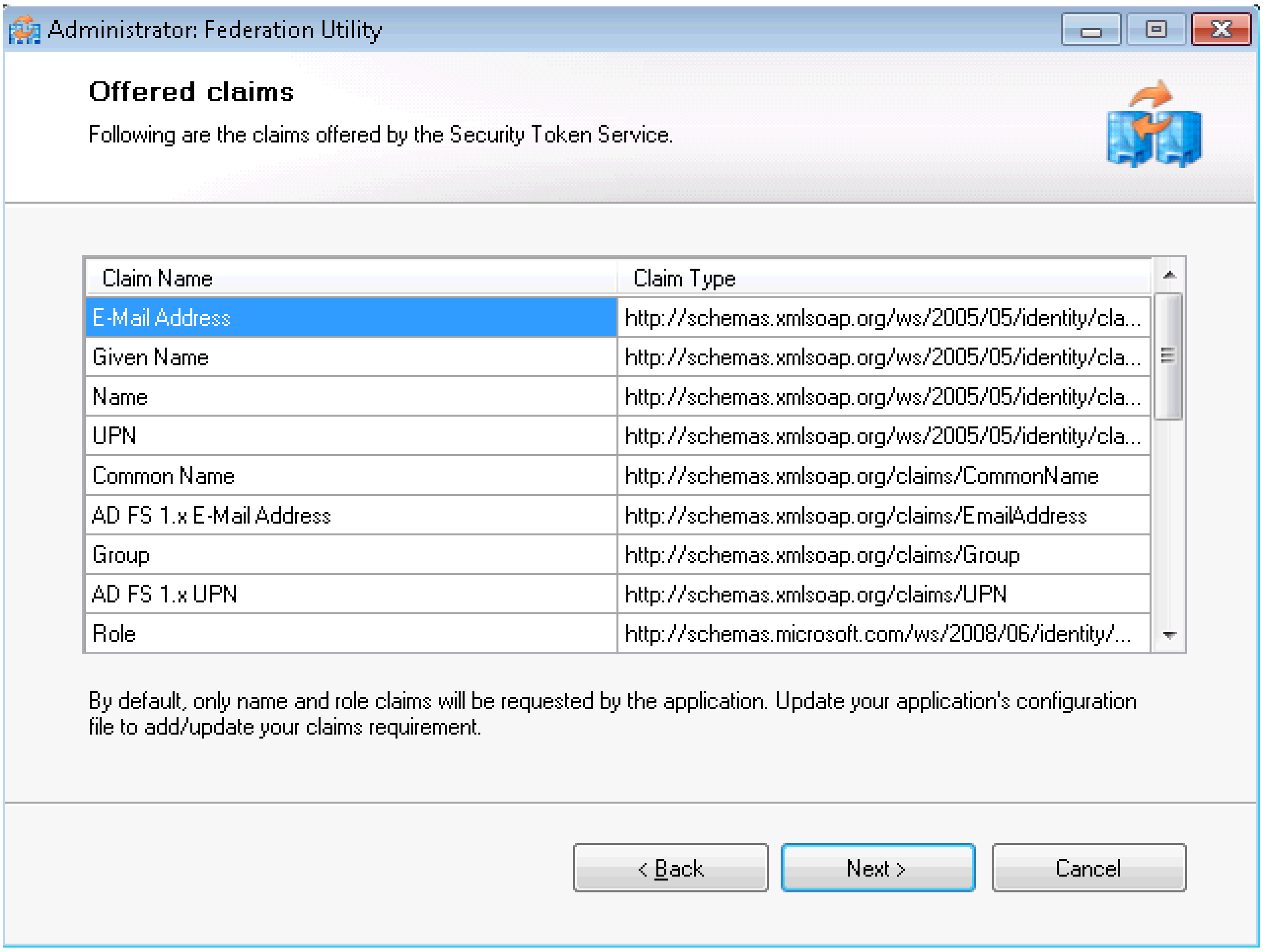
Security tokens issued by the STS will be encrypted by the selected certificate.
Note: Make sure that the private key of this encryption certificate is accessible by the Windows identity under which the application runs (example: NetworkService).

Encryption Certificate

● Generate a default certificate

○ Select an existing certificate from store

9. A list of claims offered by the STS is presented.



Offered claims

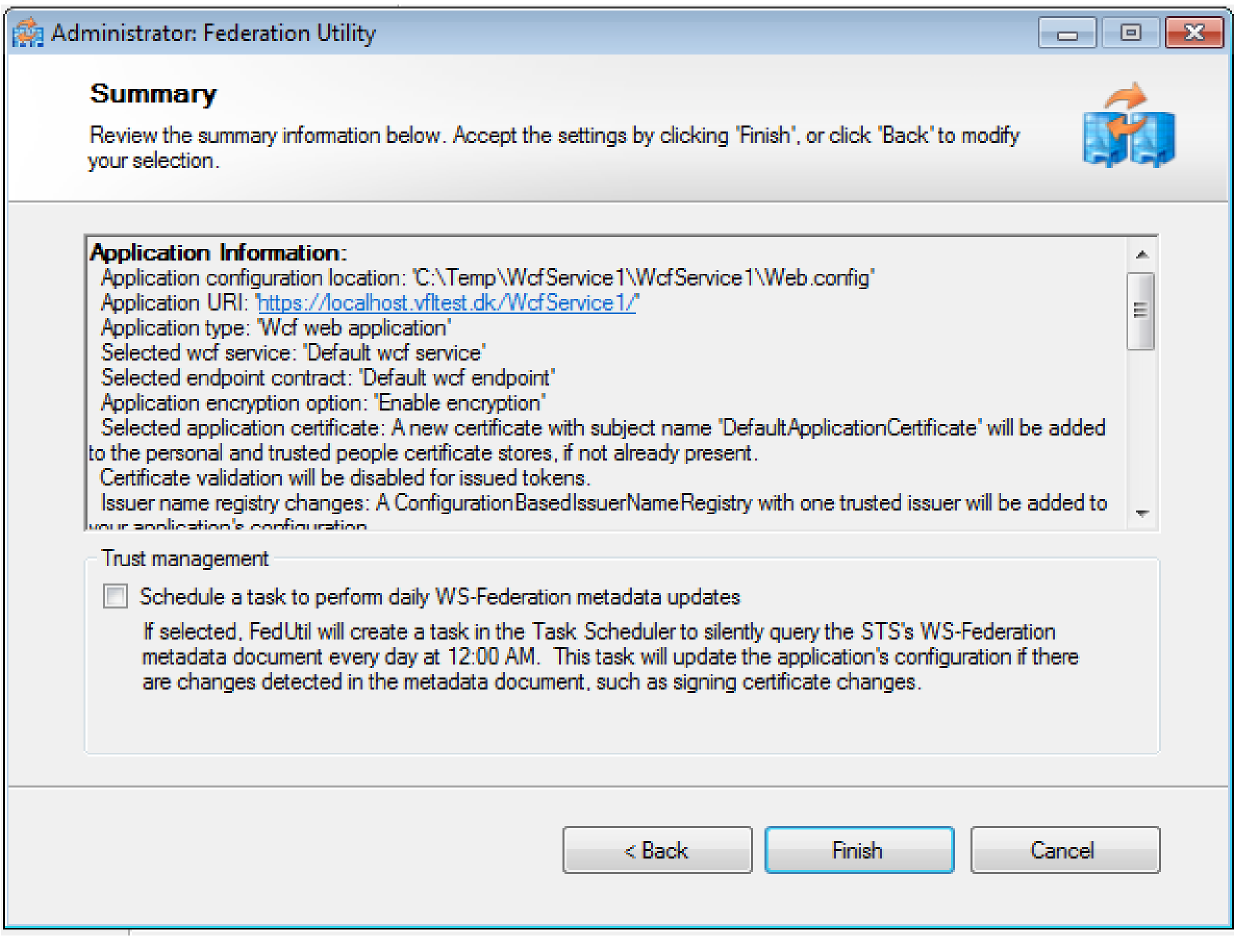
Following are the claims offered by the Security Token Service.

Claim Name	Claim Type
E-Mail Address	http://schemas.xmlsoap.org/ws/2005/05/identity/cla...
Given Name	http://schemas.xmlsoap.org/ws/2005/05/identity/cla...
Name	http://schemas.xmlsoap.org/ws/2005/05/identity/cla...
UPN	http://schemas.xmlsoap.org/ws/2005/05/identity/cla...
Common Name	http://schemas.xmlsoap.org/claims/CommonName
AD FS 1.x E-Mail Address	http://schemas.xmlsoap.org/claims/EmailAddress
Group	http://schemas.xmlsoap.org/claims/Group
AD FS 1.x UPN	http://schemas.xmlsoap.org/claims/UPN
Role	http://schemas.microsoft.com/ws/2008/06/identity/...

By default, only name and role claims will be requested by the application. Update your application's configuration file to add/update your claims requirement.

Note that this list is not maintained in the DLBR Common Login Federation, so the list is of little use (and has no bearing on the claims actually issued to the RP).

10. A summary is shown as the last step in the "FedUtil" wizard.



Summary

Review the summary information below. Accept the settings by clicking 'Finish', or click 'Back' to modify your selection.

Application Information:
Application configuration location: 'C:\Temp\WcfService1\WcfService1\Web.config'
Application URI: 'https://localhost.vfltest.dk/WcfService1/'
Application type: 'Wcf web application'
Selected wcf service: 'Default wcf service'
Selected endpoint contract: 'Default wcf endpoint'
Application encryption option: 'Enable encryption'
Selected application certificate: A new certificate with subject name 'DefaultApplicationCertificate' will be added to the personal and trusted people certificate stores, if not already present.
Certificate validation will be disabled for issued tokens.
Issuer name registry changes: A ConfigurationBasedIssuerNameRegistry with one trusted issuer will be added to your application's configuration.

Trust management

□ Schedule a task to perform daily WS-Federation metadata updates

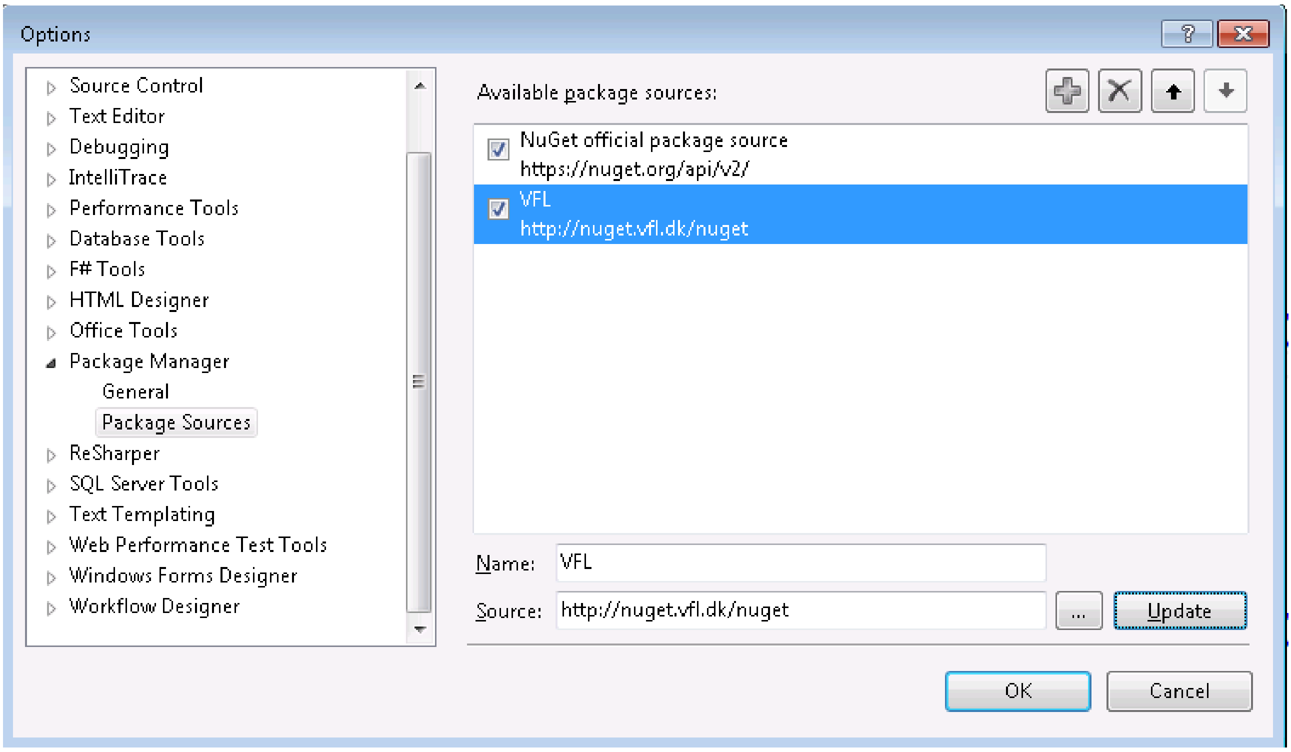
If selected, FedUtil will create a task in the Task Scheduler to silently query the STS's WS-Federation metadata document every day at 12:00 AM. This task will update the application's configuration if there are changes detected in the metadata document, such as signing certificate changes.

11. Cleanup Web.config file.
    The section with “`<claimTypeRequirements>`” may be deleted.
    The secion “`<serviceCredentials>`“ should be deleted.

    Change the security mode as shown here:
    ```
    <security mode="TransportWithMessageCredential">
    ```

    Also the message setting should be changed as follows:
    ```
    <message establishSecurityContext="false" issuedKeyType="BearerKey">
    ```

12. To simplify integration between the WCF application and the AD FS 2.0 identity/claims provider, components have been developed. To facilitate the initial plumbing, the components are available as NuGet packages in the VFL NuGet repository (http://nuget.vfl.dk/nuget). To install the packages, right-click the project in Solution Explorer and select “Manage NuGet Packages…”. In the left pane choose “VFL”, select package “Dlbr.CommonLogin.IdentityModel.WebService” and click “Install”.
If VFL package source has been enabled as available package source in the NuGet Package Manager, continue to next step. Otherwise click “Settings”, add “VFL” and “http://nuget.vfl.dk/nuget”.



Available package sources:

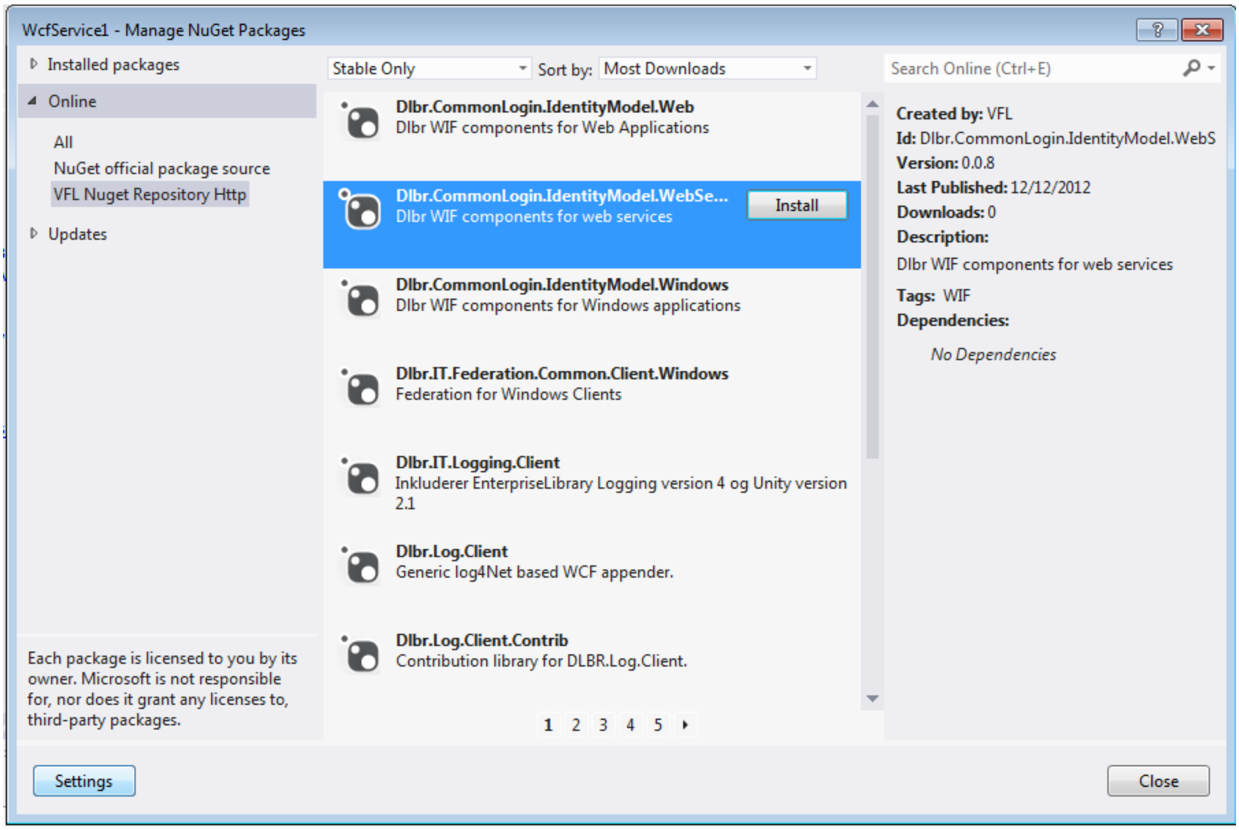
☑ NuGet official package source
https://nuget.org/api/v2/

☑ VFL
http://nuget.vfl.dk/nuget

Name: VFL

Source: http://nuget.vfl.dk/nuget

13. In Solution Explorer, right-click project file and choose menu item "Manage NuGet Package…". From the VFL store select "Dlbr.CommonLogin.IdentityModel.WebService" and click "Install".



Dlbr.CommonLogin.IdentityModel.Web
Dlbr WIF components for Web Applications

Dlbr.CommonLogin.IdentityModel.WebSe...
Dlbr WIF components for web services

Dlbr.CommonLogin.IdentityModel.Windows
Dlbr WIF components for Windows applications

Dlbr.IT.Federation.Common.Client.Windows
Federation for Windows Clients

Dlbr.IT.Logging.Client
Inkluderer EnterpriseLibrary Logging version 4 og Unity version 2.1

Dlbr.Log.Client
Generic log4Net based WCF appender.

Dlbr.Log.Client.Contrib
Contribution library for DLBR.Log.Client.

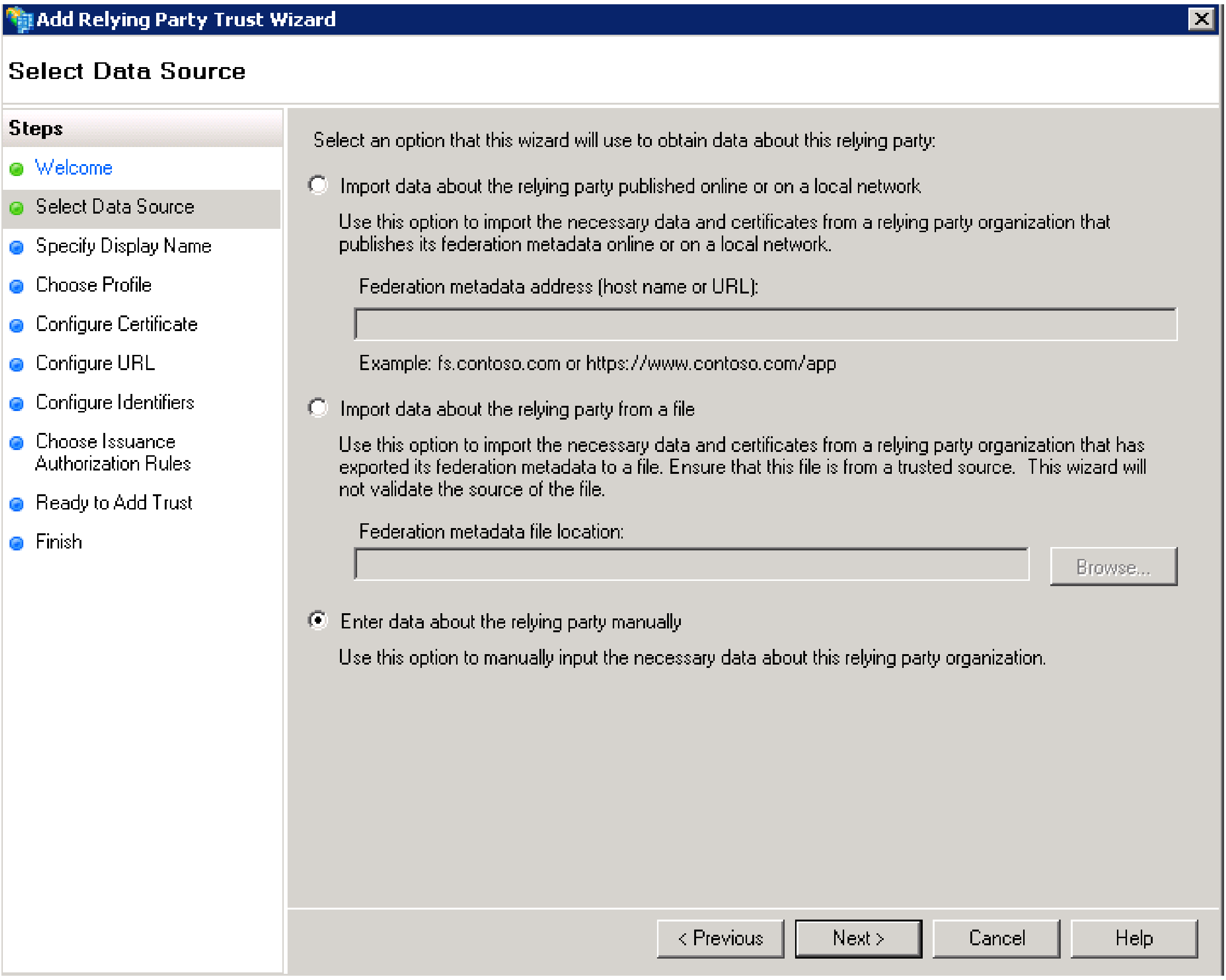
Created by: VFL
Id: Dlbr.CommonLogin.IdentityModel.WebS
Version: 0.0.8
Last Published: 12/12/2012
Downloads: 0
Description:
Dlbr WIF components for web services
Tags: WIF
Dependencies:
No Dependencies

Each package is licensed to you by its owner. Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.

2.4 Step 4 – Add STS Relying Party Trust

In this step we will add a new Relying Party Trust to the STS configuration.

1. Login to dev-idp.vfltest.dk or devtest-idp.vfltest.dk server, depending on whether the application must federate with DEV or DEVTEST identity provider.
2. In the “Administrative Tools” menu select “AD FS 2.0 Management”.
3. In the “Actions” pane to the right, choose “Add Relying Party Trust…”.
4. Select “Enter data about the relying party manually”.



Select an option that this wizard will use to obtain data about this relying party:

○ Import data about the relying party published online or on a local network

Use this option to import the necessary data and certificates from a relying party organization that publishes its federation metadata online or on a local network.

Federation metadata address (host name or URL):

Example: fs.contoso.com or https://www.contoso.com/app

○ Import data about the relying party from a file

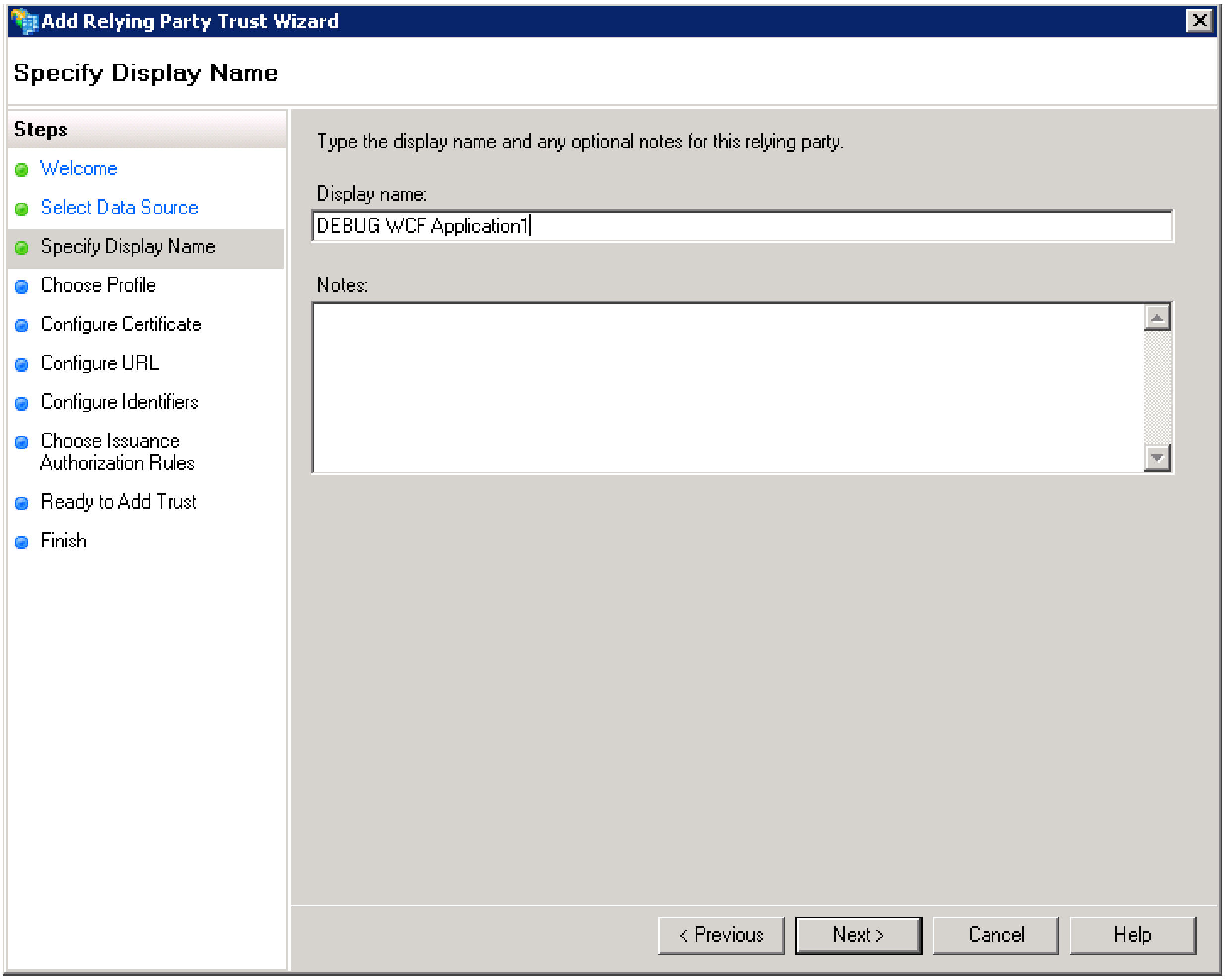
Use this option to import the necessary data and certificates from a relying party organization that has exported its federation metadata to a file. Ensure that this file is from a trusted source. This wizard will not validate the source of the file.

Federation metadata file location:

● Enter data about the relying party manually

Use this option to manually input the necessary data about this relying party organization.

5. Enter a display name, this value is purely informational, so any value will do. However you should follow the naming convention used in all the other relying parties, i.e. the Display name should start with “DEBUG”, “DEV”, “DEVTEST” or the like depending on the environment it should support.



Add Relying Party Trust Wizard

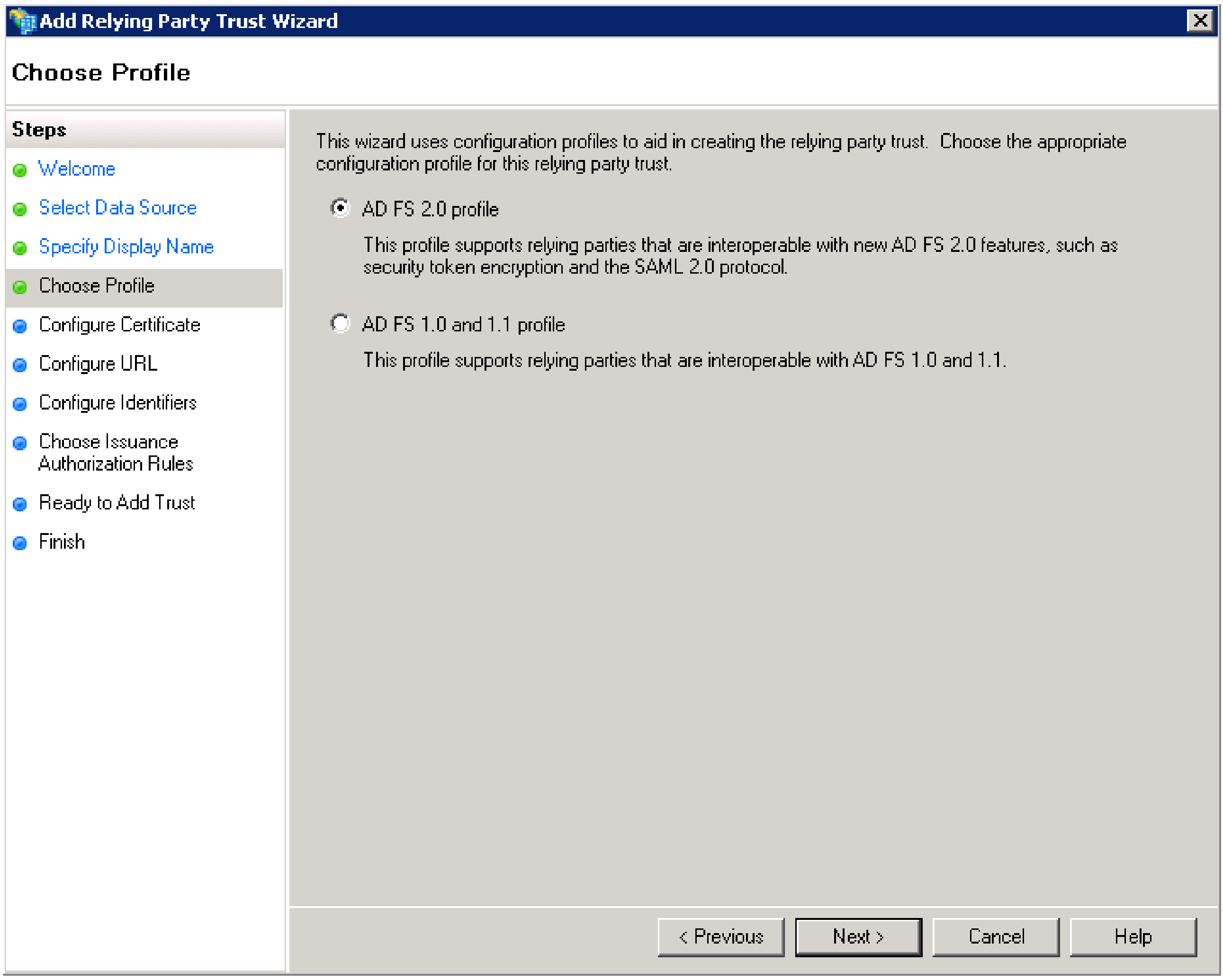
Specify Display Name

Type the display name and any optional notes for this relying party.

Display name: DEBUG WCF Application1

Notes:

6. Select AD FS 2.0 profile



Add Relying Party Trust Wizard

Choose Profile

This wizard uses configuration profiles to aid in creating the relying party trust. Choose the appropriate configuration profile for this relying party trust.

● AD FS 2.0 profile

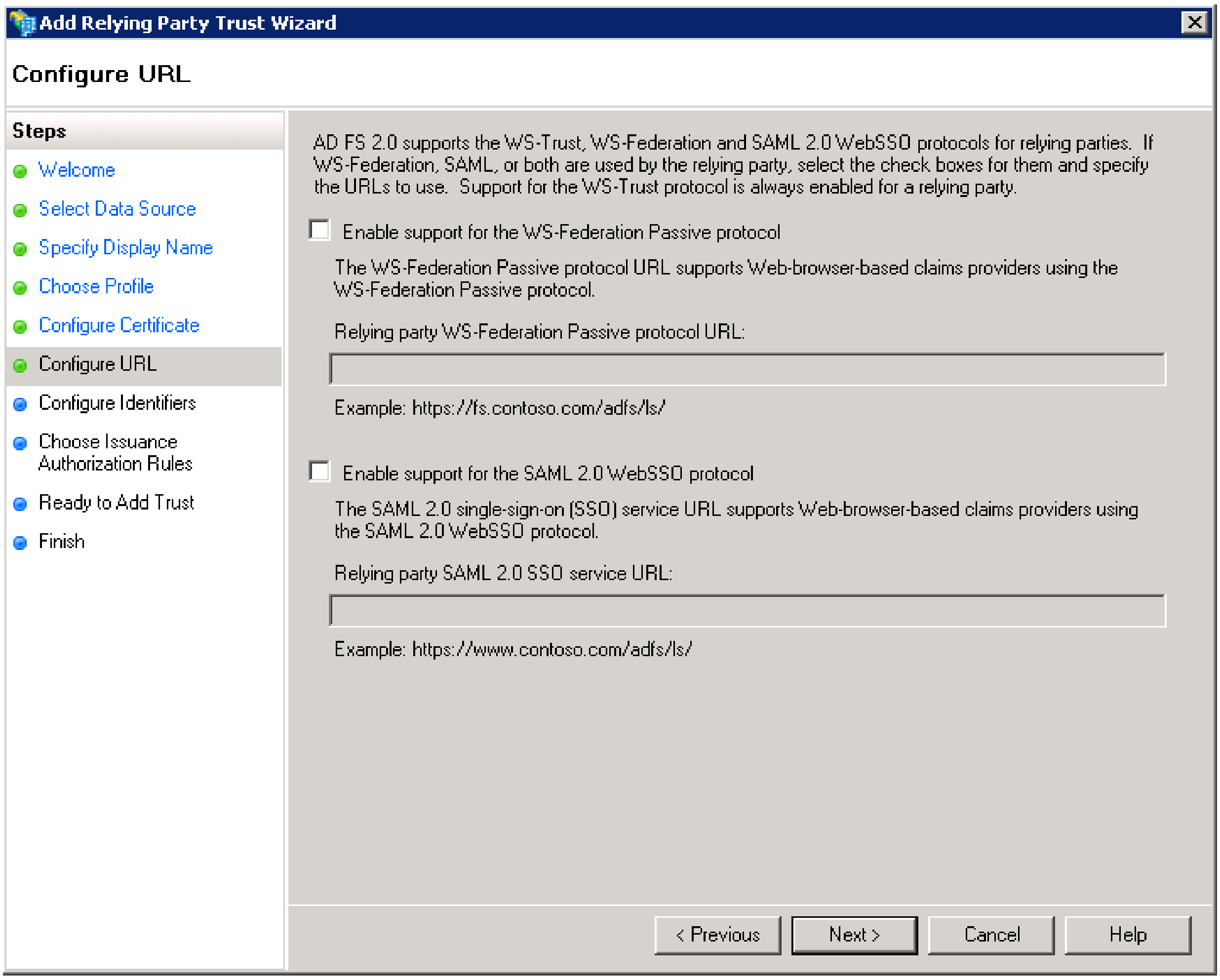
This profile supports relying parties that are interoperable with new AD FS 2.0 features, such as security token encryption and the SAML 2.0 protocol.

○ AD FS 1.0 and 1.1 profile

This profile supports relying parties that are interoperable with AD FS 1.0 and 1.1.

7. Leave “Configure Certificate”, security tokens will not be encrypted.

8. Leave the two checkboxes empty:



Add Relying Party Trust Wizard

Configure URL

AD FS 2.0 supports the WS-Trust, WS-Federation and SAML 2.0 WebSSO protocols for relying parties. If WS-Federation, SAML, or both are used by the relying party, select the check boxes for them and specify the URLs to use. Support for the WS-Trust protocol is always enabled for a relying party.

□ Enable support for the WS-Federation Passive protocol

The WS-Federation Passive protocol URL supports Web-browser-based claims providers using the WS-Federation Passive protocol.

Relying party WS-Federation Passive protocol URL:

Example: https://fs.contoso.com/adfs/ls/

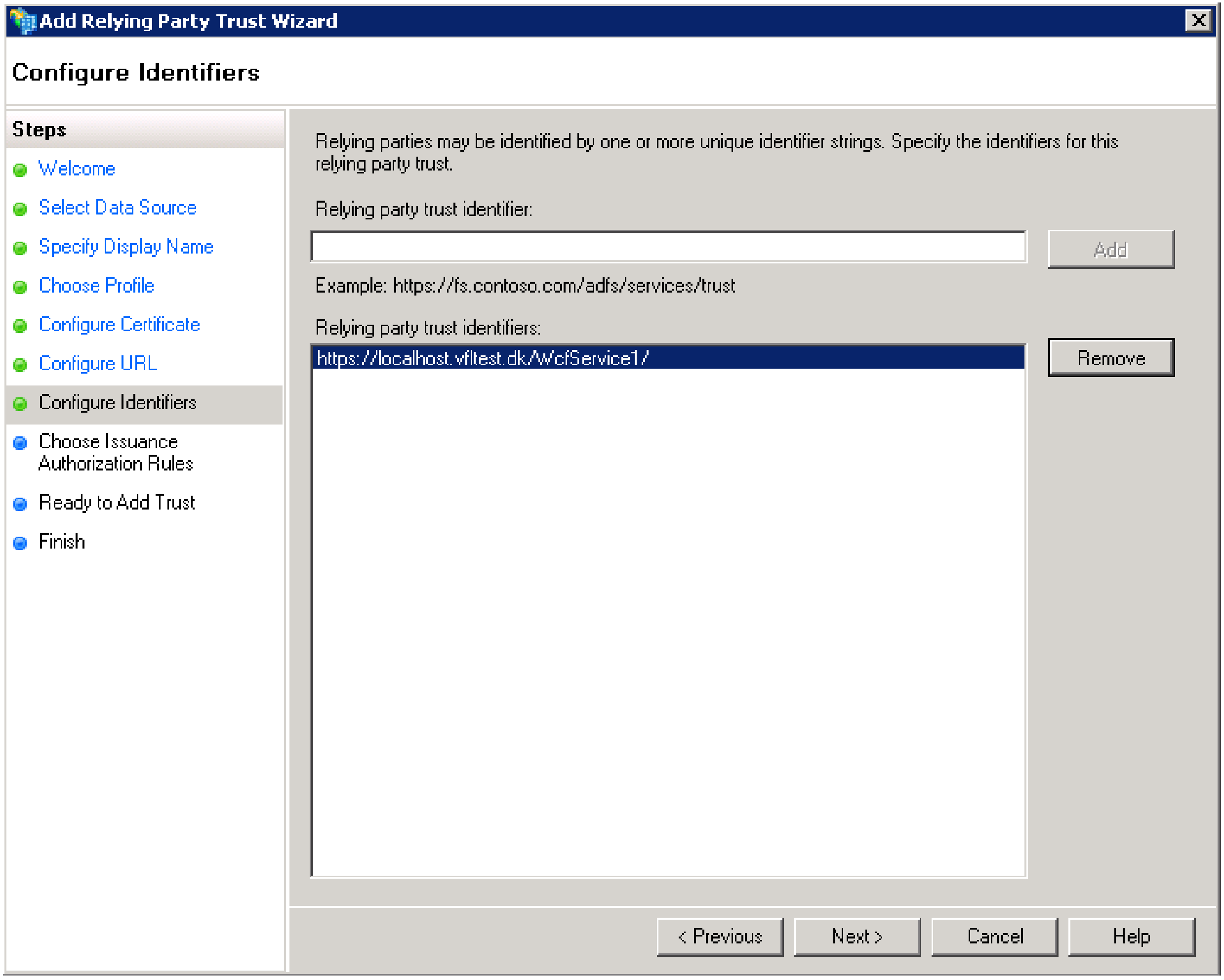
□ Enable support for the SAML 2.0 WebSSO protocol

The SAML 2.0 single-sign-on (SSO) service URL supports Web-browser-based claims providers using the SAML 2.0 WebSSO protocol.

Relying party SAML 2.0 SSO service URL:

Example: https://www.contoso.com/adfs/ls/

9. In “Configure Identifiers”, you must enter your applications identifier. Typically you would want to use the same URL as your application is configured to use, e.g. “https://localhost.vfltest.dk/WcfService1/”. Remember the last ‘/’. Press Add before moving on to the next step.



Add Relying Party Trust Wizard

Configure Identifiers

Steps
- Welcome
- Select Data Source
- Specify Display Name
- Choose Profile
- Configure Certificate
- Configure URL
- Configure Identifiers
- Choose Issuance Authorization Rules
- Ready to Add Trust
- Finish

Relying parties may be identified by one or more unique identifier strings. Specify the identifiers for this relying party trust.

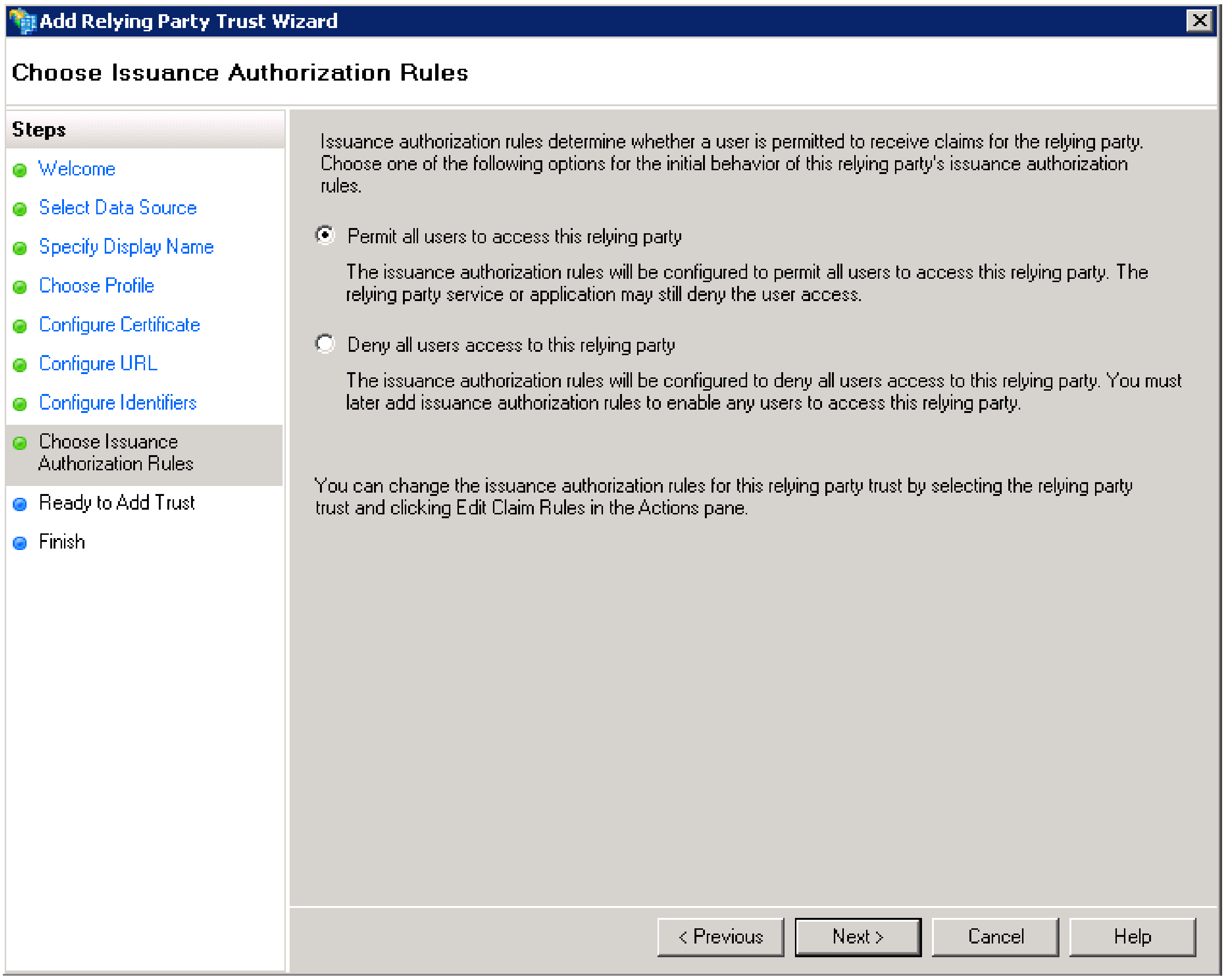
Relying party trust identifier:

Example: https://fs.contoso.com/adfs/services/trust

Relying party trust identifiers:

https://localhost.vfltest.dk/WcfService1/

10. Select “Permit all users to access the relying party”.



Add Relying Party Trust Wizard

Choose Issuance Authorization Rules

Issuance authorization rules determine whether a user is permitted to receive claims for the relying party. Choose one of the following options for the initial behavior of this relying party's issuance authorization rules.

● Permit all users to access this relying party

The issuance authorization rules will be configured to permit all users to access this relying party. The relying party service or application may still deny the user access.

○ Deny all users access to this relying party

The issuance authorization rules will be configured to deny all users access to this relying party. You must later add issuance authorization rules to enable any users to access this relying party.

You can change the issuance authorization rules for this relying party trust by selecting the relying party trust and clicking Edit Claim Rules in the Actions pane.

11. Wizard step “Ready to Add Trust” is informational so just continue.

12. In the last step click “Close” button. If “Open the Edit Claim Rules dialog for this relying party…..” is checked, the “Edit Claim Rules” dialog opens when finishing the wizard.

    A note about relying party WCF application hosted on Windows Server 2003. When the security token containing claims, that is issued by the Security Token Service (STS) upon a successfull authentication of the user, is issued, it is digitally signed by the STS signing certificate. By default this signing is based on the SHA-256 hash algorithm. For the relying party to read the security token, the SHA-256 hashing algorithm must be installed on the server hosting the web application, for the federation process between the STS and the relying party to function.

    On a Windows Server 2003 the SHA-256 hashing algorithm is not installed by default. Source code is available that enable the SHA-256 hashing algorithm. In Visual Studio create a .NET 4.0 console project and add the file from TFS: $/DLBRLogin/DLBRLogin/trunk/Tools/SHA-256. How-to instructions are available in the source code.

13. Note that only 2 claims is issued by default when a user is authenticated by the IdP. These claims (claimtype) are http://schemas.microsoft.com/ws/2008/06/identity/claims/authenticationmethod and http://schemas.microsoft.com/ws/2008/06/identity/claims/authenticationinstant, describing how and when the user was authenticated. Examples of values are http://schemas.microsoft.com/ws/2008/06/identity/authenticationmethod/password (if the user was authenticated by a password) and “2012-12-18T13:09:54.814Z” accordingly.

    A Claims Rule Language example of how to issue other claim types, such as Active Directory groups membership and Windows logon user id, is shown in Appendix 2.

14. Verify the new WCF Service application by adding e.g. a simple console app to your solution. Add the “Dlbr.CommonLogin.IdentityModel.Windows” NuGet package to this project, and type some simple client code as for instance this:

```
SecurityToken securityToken = AdfsHelper.GetSecurityToken(
  "https://devtest-idp.vfltest.dk/adfs/services/trust/13/usernamemixed",
  "https://localhost.vfltest.dk/WcfService1/", "DCFIntegrationstest",
  "dcftest");

  // Act
  using (var service = new WcfServiceWrapper<IService1>(securityToken,
         "https://localhost.vfltest.dk/WcfService1/Service1.svc"))
  {
      string data = service.Channel.GetData(15);
      Console.WriteLine("Hentet data: {0}", data);
  }
```

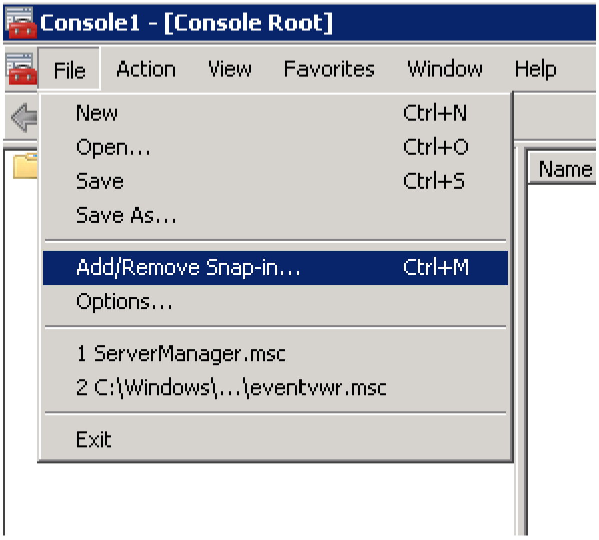
3 Appendix 1

3.1 Manually configuring IIS

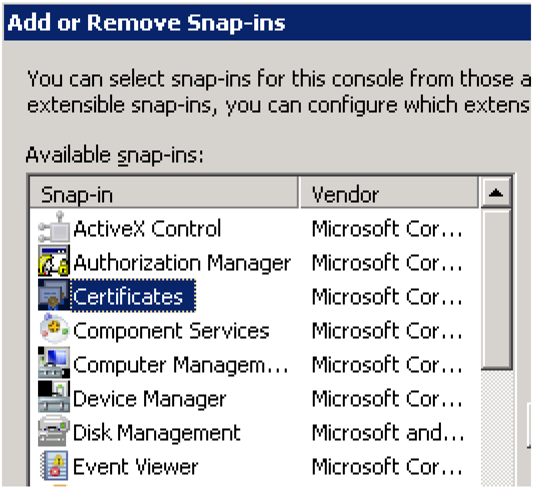
Instead of using the PowerShell script “CreateWebsite.ps1” to facilitate creation of IIS application pool and web site, a new web site can manually be configured using “Internet Information Services Manager”. Be aware that SSL certificate with subject “*.vfltest.dk” must be installed in store “LocalMachine\My” prior to creating the web site.

Installation of the SSL certificate can be done using “Microsoft Management Console”:

1. In the Windows start menu select “Run…”, type mmc and click OK.
2. In MMC, choose “File – Add/Remove Snap-in…”



3. Double-click on “Certificates”

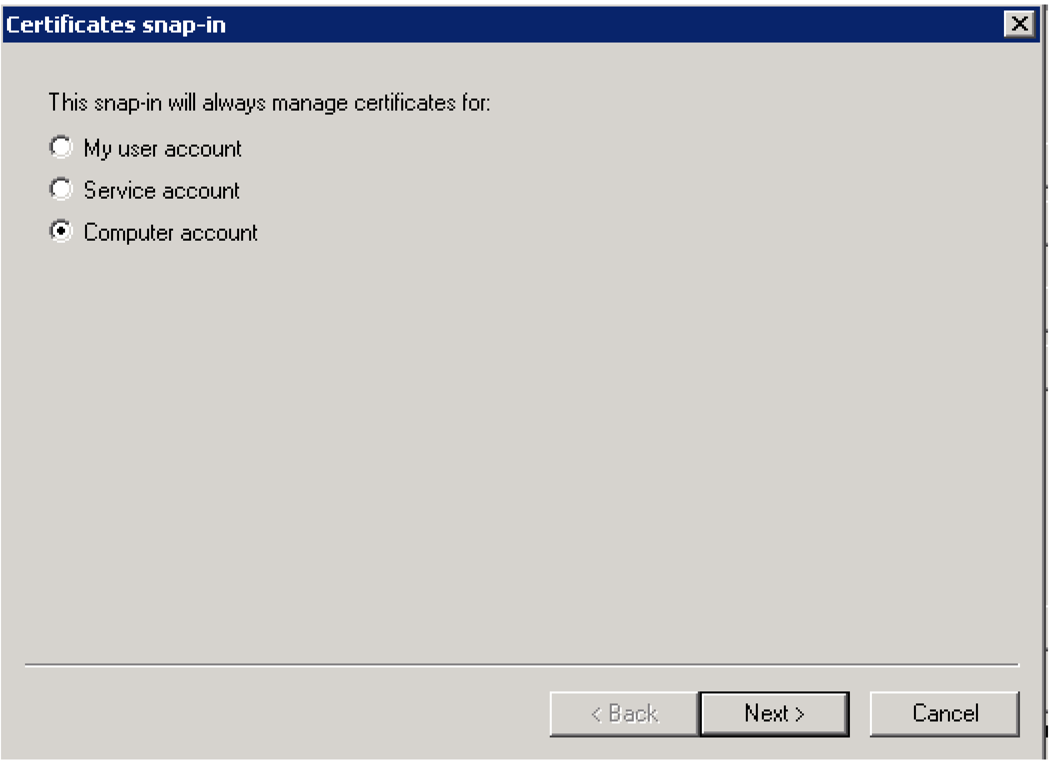


You can select snap-ins for this console from those a
extensible snap-ins, you can configure which extens

Available snap-ins:

Snap-in	Vendor
ActiveX Control	Microsoft Cor...
Authorization Manager	Microsoft Cor...
Certificates	Microsoft Cor...
Component Services	Microsoft Cor...
Computer Managem...	Microsoft Cor...
Device Manager	Microsoft Cor...
Disk Management	Microsoft and...
Event Viewer	Microsoft Cor...

4. Select “Computer account”, click “Next”, “Finish” and “OK”



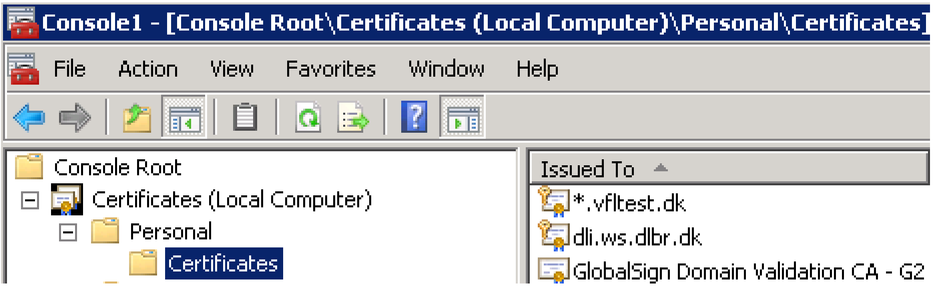
This snap-in will always manage certificates for:

○ My user account

○ Service account

● Computer account

5. Expand “Certificates – Personal – Certificates”. Right-click the “Certificates” folder and choose “All tasks – Import…”. In the wizard, select file type “Personal Information Type (*.pfx)” and select the file to be imported. When prompted, type the password for the private key.



Console1 - [Console Root\Certificates (Local Computer)\Personal\Certificates]
File
Action
View
Favorites
Window
Help
Console Root
Certificates (Local Computer)
Personal
Certificates
Issued To

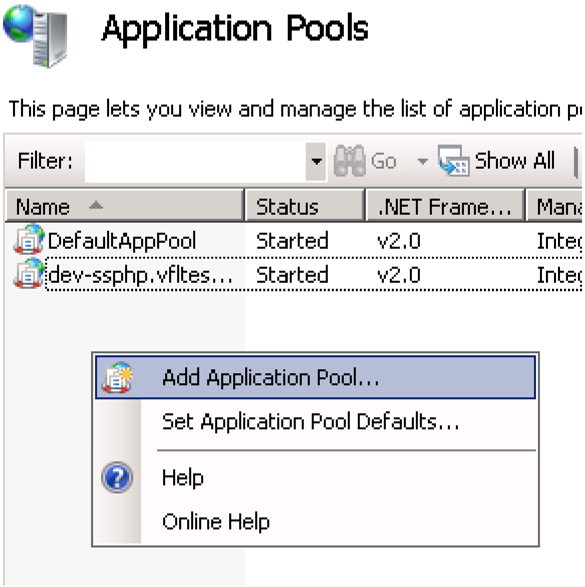
*.vfltest.dk

dli.ws.dlbr.dk

GlobalSign Domain Validation CA - G2

Configuring IIS application pool and web site can be done using “Internet Information Services (IIS) Manager”:

1. After installation of the SSL certificate, it’s time to create the web site, which is accomplished in “Internet Information Services (IIS) Manager (found in Start – Administrative Tools). Start with creating a new application pool by clicking on “Application Pools” in the left pane:



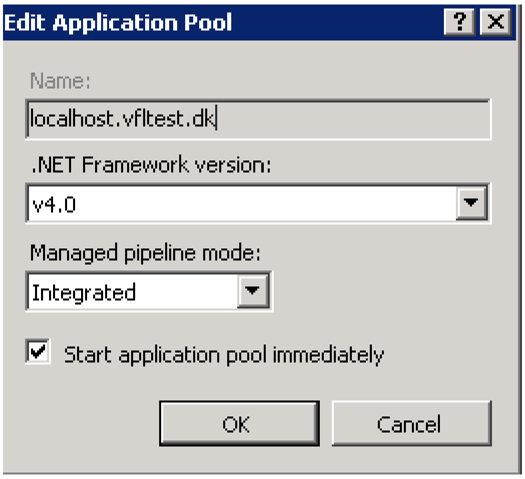
Application Pools

This page lets you view and manage the list of application p

Filter:

Name	Status	.NET Frame...	Mana
DefaultAppPool	Started	v2.0	Integ
dev-ssphp.vfltes...	Started	v2.0	Integ

2. Configure it to use .Net Framework version “4.0” and set Managed pipeline mode to “Integrated”.



Edit Application Pool

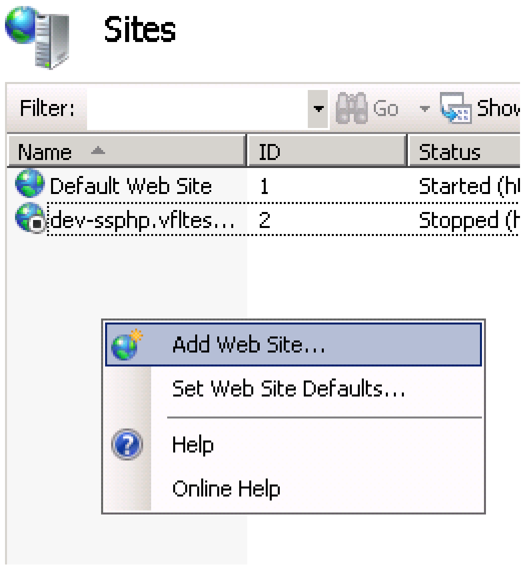
Name: localhost.vfltest.dk

.NET Framework version: v4.0

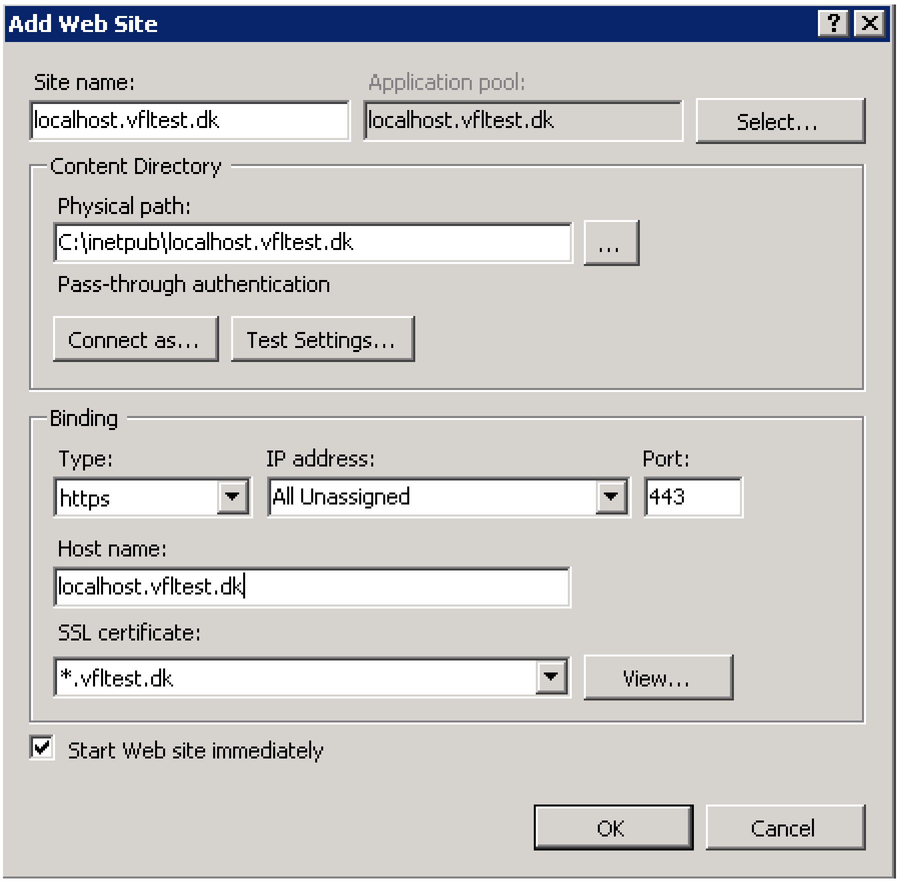
Managed pipeline mode: Integrated

☑ Start application pool immediately

3. Create a new web site by clicking on “Sites” in the left pane



4. Configure it to use the previously created application pool. Choose “https” binding and select “*.vfltest.dk” as SSL certificate.



Add Web Site

Site name: localhost.vfltest.dk

Application pool: localhost.vfltest.dk

Content Directory

Physical path: C:\inetpub\localhost.vfltest.dk

Pass-through authentication

Binding

Type: https

IP address: All Unassigned

Port: 443

Host name: localhost.vfltest.dk

SSL certificate: *.vfltest.dk

☑ Start Web site immediately

4 Appendix 2

4.1 How to issue Windows account name (logon user id) as a claim

1. Login to dev-idp.vfltest.dk or devtest-idp.vfltest.dk server, depending on whether the application must federate with DEV or DEVTEST identity provider.
2. Open “Start – Administrative Tools – AD FS 2.0 Management”.
3. Expand “Trust Relationships – Relying Party Trusts”, right-click the relevant relying party trust registration and choose “Edit Claim Rules…”.
4. Click on “Add Rule…” and select “Pass Through or Filter an Incoming Claim”.
5. Enter a Claim Rule Name. The value is optional.
6. In the “Incoming Claim type” drop-down box select “Windows account name”.
7. Select “Pass through all claim values” and click “Finish” button. Note that the format of Windows account name is “domain\userid”, e.g. PROD\LCMCM.

4.2 How to issue group membership as claims

1. Execute steps 1-3 in section 4.1.
2. Click on “Add Rule…” and select “Send LDAP attributes as Claims”.
3. Enter a Claim Rule Name. The value is optional.
4. In the “Attribute store” drop-down box select “Active Directory”.
5. In the “LDAP Attribute” drop-down box select “Token Groups – Unqualified Names”.
6. In the “Outgoing Claim Type” drop-down box select “Role”.

4.3 How to issue specific group membership as custom claims with DCF groups as an example

1. Execute steps 1-3 in section 4.1.
2. Click on “Add Rule…” and select “Send Claims Using a Custom Rule”.
3. Enter a Claim Rule Name. The value is optional.
4. Add the following as “Custom Rule”:

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

   This rule, based on the “windowsaccountname” for the user, adds all groups to the type “TokenGroups”.

5. Add the following as a new “Custom Rule”:
   c:[Type == "TokenGroups", Value =~ "^(?i)GTALCDCF"] => issue(Type = "http://dcf.ws.dlbr.dk/ws/2008/04/authorization/claims/serviceauthorizations", Issuer = c.Issuer, OriginalIssuer = c.OriginalIssuer, Value = regexreplace(c.Value, "^GTALC", ""), ValueType = c.ValueType);
   This rule selects groups from “TokenGroups”, based on group name starting with “GTALCDCF”. Finally “GTALC” is stripped from group name making all groups starting with “DCF”.
   Note that the order of adding (executing) these rules is fixed as the rule in this section uses output from the rule in the previous section (4.3.4).
6. To make the role claims available in the web application, add the “roleClaimType” to web.config in section “samlSecurityTokenRequirement”

```
<samlSecurityTokenRequirement….
…..
<roleClaimType value= http://dcf.ws.dlbr.dk/ws/2008/04/authorization/claims/servic
eauthorizations />
</samlSecurityTokenRequirement>
```

   Note that http://dcf.ws.dlbr.dk/ws/2008/04/authorization/claims/serviceauthorizations is a custom claim type (i.e. not one of the standard claim types issued by AD FS 2.0 out-of-the-box).
7. Add the following C# code to the application to iterate the claims:

```
        protected void Page_Load(object sender, EventArgs e)
        {
            Response.Write(DumpClaims());
        }

        private string DumpClaims()
        {
            var principal = (ClaimsPrincipal)Thread.CurrentPrincipal;
            var identity = (IClaimsIdentity)principal.Identity;

            var result = new StringBuilder();
            var level = "Identity";
            while (identity != null)
            {
                var claimStrings =
                    identity.Claims.Select(
                        claim =>

string.Format("<tr><td>{0}</td><td>{1}</td><td>{2}</td><td>{3}</td><td>{4}</td></
tr>",
                                      claim.ClaimType, claim.Issuer,
claim.OriginalIssuer, claim.Subject, claim.Value));

                var formattedClaimsForIdentity = level + "<br /><table
border='1'><tr><td>Claimtype</td><td>Issuer</td><td>OriginalIssuer</td><td>Subjec
t</td><td>Value</td></tr> " + string.Join("\n", claimStrings) + "</table>";
                result.AppendLine(formattedClaimsForIdentity);
                identity = identity.Actor;
                level = level + ".Actor";
            }
            return result.ToString();
        }
```

   Note that the code is not required for the application to execute. It has only informational value and can be utilized in debugging scenarios.

4.4 How to issue Name id as a claim

1. Execute steps 1-3 in section 4.1.
2. Click on “Add Rule…” and select “Send LDAP attributes as Claims”.
3. Enter a Claim Rule Name. The value is optional.
4. In the “Attribute store” drop-down box select “Active Directory”.
5. In the “LDAP Attribute” drop-down box select “SAM-Account-Name”.
6. In the “Outgoing Claim Type” drop-down box select “Name ID”.
7. To make the name id available in the web application, add the “nameClaimType” to web.config in section “samlSecurityTokenRequirement”

```
<samlSecurityTokenRequirement….
…..
<nameClaimType value=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameid
entifier />
</samlSecurityTokenRequirement>
```

8. The name id can be read from the property “`Thread.CurrentPrincipal.Identity.Name`”.