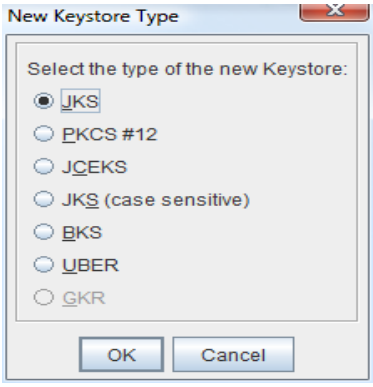
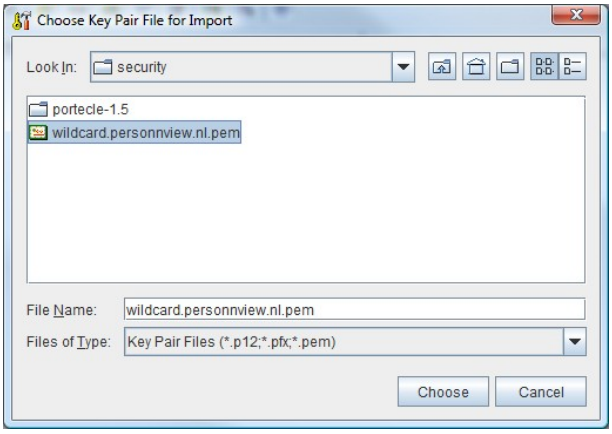
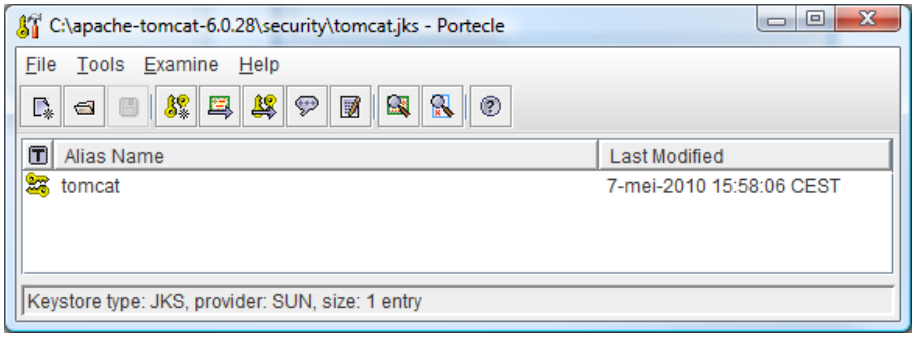
Configure SSL on Tomcat

Make a security directory (for example C:\apache-tomcat-6.0.28\security) and import your SSL certificate in a keystore using the tool Portecle (https://sourceforge.net/projects/portecle) and name it 'tomcat'. Create a JKS keystore and remember secure it with a password;





Import your SSL certificate;



Edit your Tomcat server.conf and add the following part to enable SSL on your server. If you use a reverse-proxy server to handle your SSL certificate, you can skip this SSL configuration altogether;

```
<Connector port="8443" maxHttpHeaderSize="8192" protocol="HTTP/1.1" SSLEnabled="true"
        maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
        enableLookups="false" disableUploadTimeout="true"
        acceptCount="100" scheme="https" secure="true"
        clientAuth="want"
        keystoreFile="C:/apache-tomcat-6.0.28/security/tomcat.jks"
        keystorePass="password"
        truststoreFile="C:/apache-tomcat-6.0.28/security/tomcat.jks"
        truststorePass="password"
        sslProtocol="TLS"
        />
```

Configure Picketlink;

See the Picketlink documentation on how to configure your web.xml and the two picketlink-handler.xml and picketlink-idfex.xml;

In this example test5.nietdus.nl is your ADFSv2 server and webtest.personnelview.nl is running your Tomcat application.

```
<PicketLinkSP
    xmlns="urn:picketlink:identity-federation:config:1.0"
    CanonicalizationMethod="http://www.w3.org/2001/10/xml-exc-c14n#"
          ServerEnvironment="tomcat">

          <!-- The URL of the Microsoft AD FS v2 IDP Server to which we post our SAML AuthnRequest -->
          <!-- ADFSv2 only supports requests to HTTPS -->
          <IdentityURL>https://test5.nietdus.nl/adfs/ls/</IdentityURL>

          <!-- The URL to our SSO servlet to which the IDP/ADFSv2 posts the SAML IDP Response -->
          <!-- ADFSv2 only support posts to HTTPS so we have to configure Tomcat for SSL - see the Tomcat server.xml-->
          <ServiceURL>https://webtest.personnelview.nl:8443/PV_Web/s/sso</ServiceURL>

          <!-- Only trust IDP SAML Responses from the following IDP domains -->
          <Trust>
                   <Domains>personnelview.nl</Domains>
          </Trust>

          <KeyProvider ClassName="org.picketlink.identity.federation.core.impl.KeyStoreKeyManager">

          <!-- Path to keystore of certificates -->
          <Auth Key="KeyStoreURL" Value="C:/apache-tomcat-6.0.28/security/composer5_keystore.jks" />
          <Auth Key="KeyStorePass" Value="store123" />

          <!-- Which certificate in the keystore do we use ourself for signing the SAML AuthnRequest to the IDP? -->
          <Auth Key="SigningKeyAlias" Value="sp_sign_cert" />
          <Auth Key="SigningKeyPass" Value="pass123" />

          <!-- Every SAML Response from the IDP is/mustbe signed and the signing must be checked to makeu
          use the IDP can be trusted -->
          <!-- Key=Domain name for which this certificate can be used to check the signing -->
          <!-- Value=Aliasname in keystore -->
          <ValidatingAlias Key="test5.nietdus.nl" Value="idp_test5.nietdus.nl" />

          </KeyProvider>

</PicketLinkSP>
```

In the following chapters you are going to create the composer5_keystore.jks keystore file with the self signed ADFSv2 IDP certificate and the self signed Picketlink SP certificate.

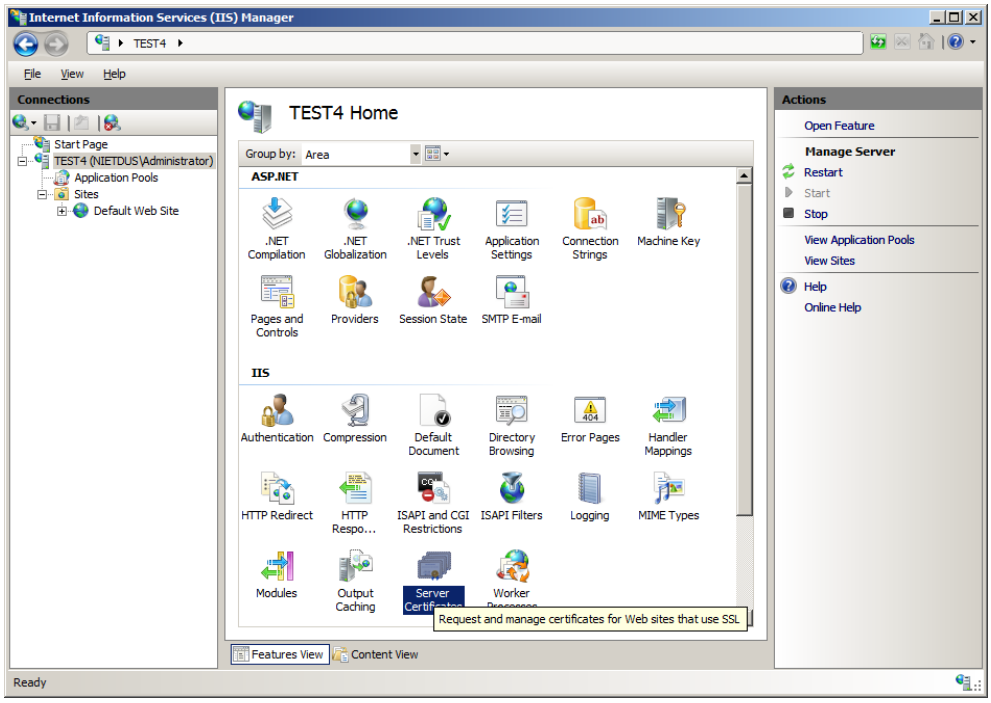
We use a IDP role named composer5 to enable/disable users from using our application. We pass the groups from the Microsoft Active Directory as an SAML role so we can test within our application to allow only users with this group. To do this you have to edit the web.xml file to protect your servlet so only requests with role composer5 are being pass on to your servlet. Our this example our servlet is called has a url pattern of /s/sso ;

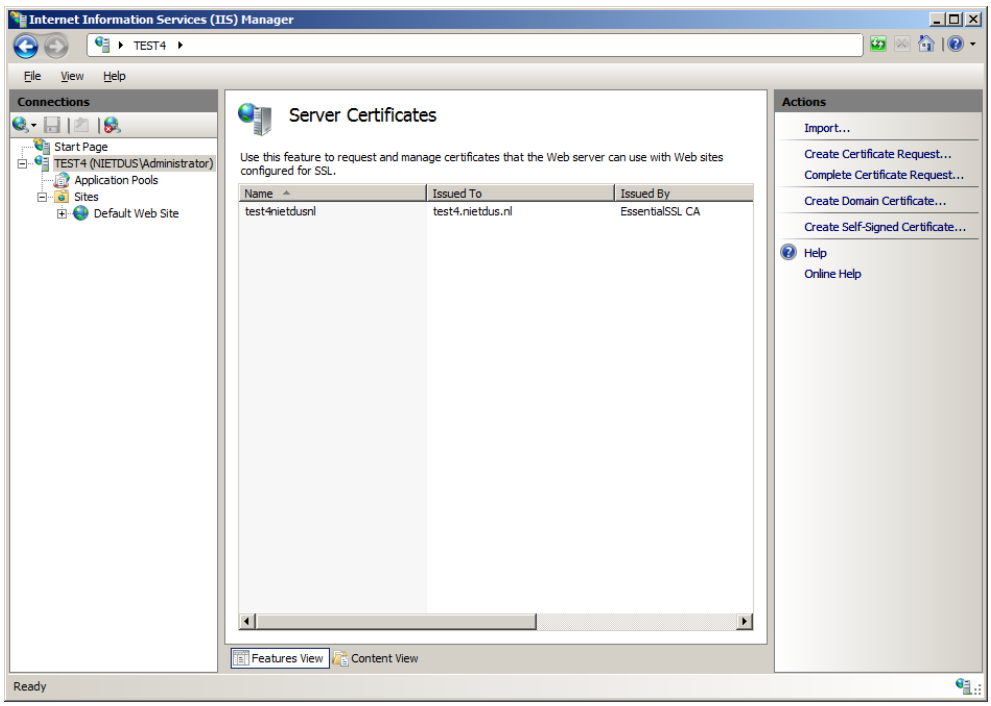
```
        <!-- Define a Security Constraint on this Application for SSO -->
        <security-constraint>
                <web-resource-collection>
                        <web-resource-name>Composer 5 Application</web-resource-name>
                        <url-pattern>/s/sso</url-pattern>
                </web-resource-collection>
                <auth-constraint>
                        <role-name>composer5</role-name>
                </auth-constraint>
        </security-constraint>

    <!-- Security roles defined on user from IDP referenced by this web application for SSO -->
    <security-role>
        <description>The role that is required to log in to the Composer Application</description>
        <role-name>composer5</role-name>
    </security-role>
```

Installatie en configuratie ADFS v2

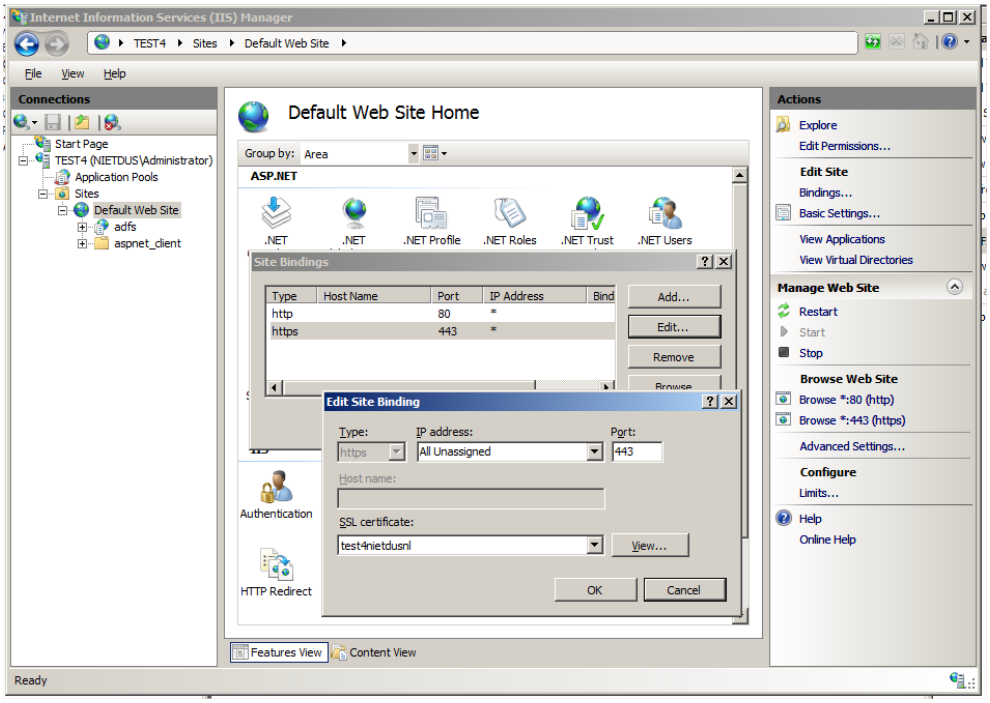
Installeer eerst ADFSv2 op de server: run de adfsv2 setup.exe van Microsoft

Configureer nu IIS met een geldig certificaat van het domain waarop de Identify Provider (IDP) moet gaan werken. Doe dit door een al reeds bestaand certificate te importen of een nieuwe aan te maken met een certificate request (CSR) en daarna het inlezen van het certificate van een certificate authority:





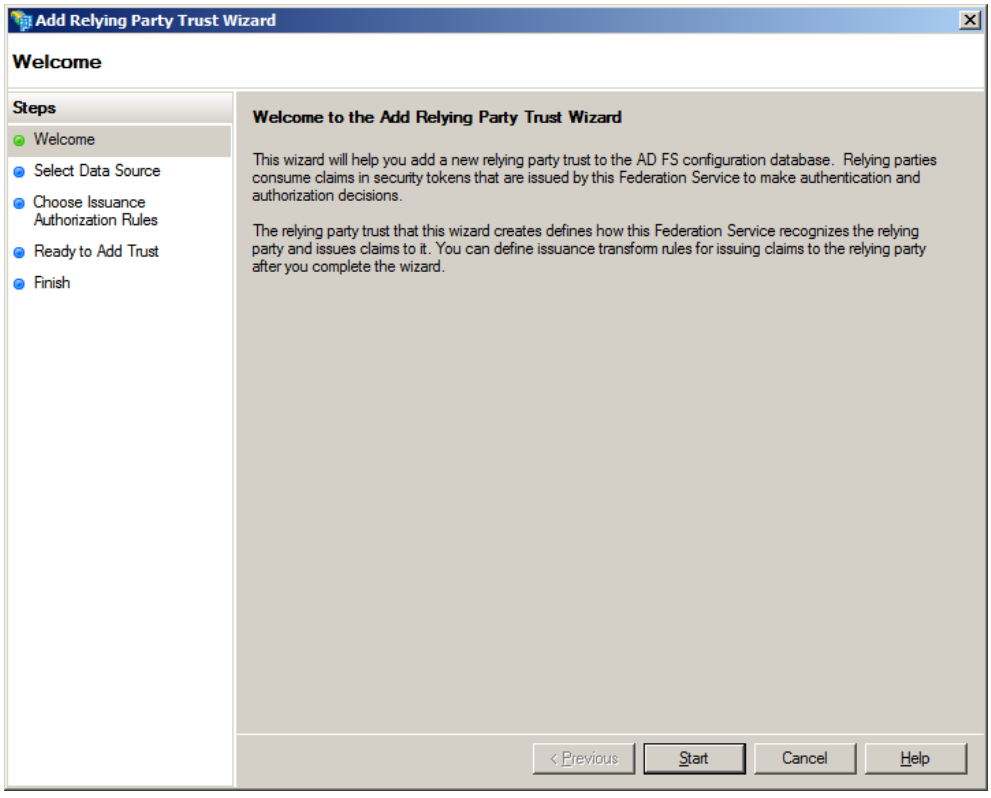
Koppel het certificaat aan de https binding van de default website waarop het adfs script staat:



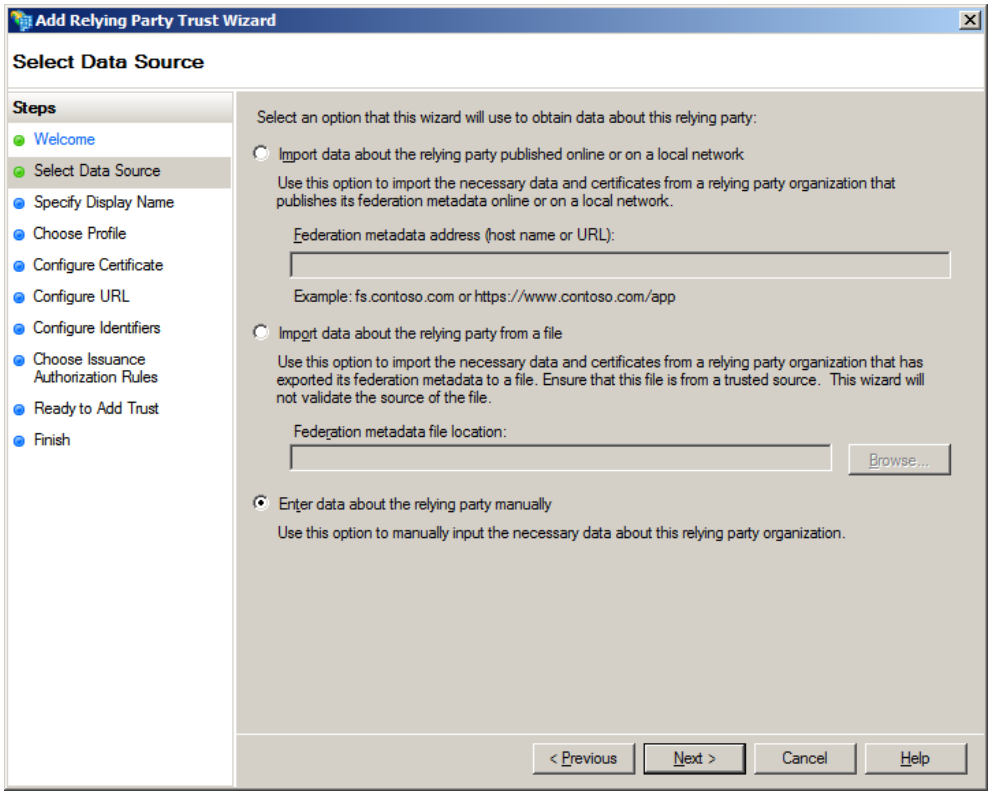
Draai nu de ADFS configuratie wizard.

next->next->next etc.

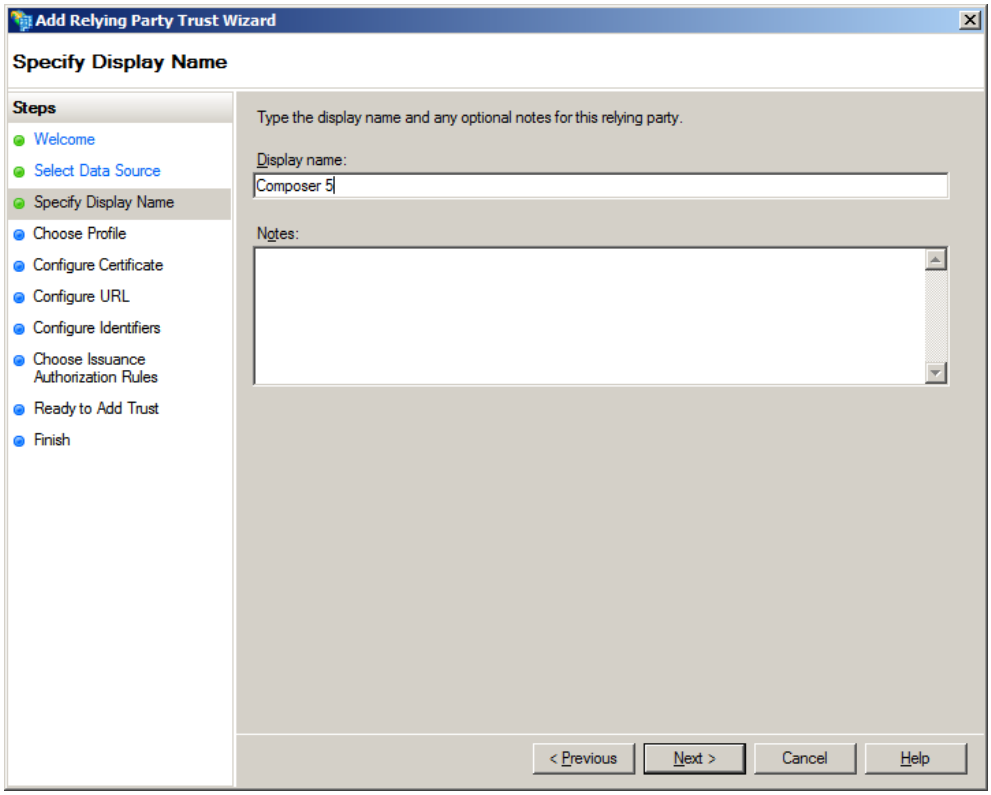
Hierna creeer een nieuwe Relying Trust Party:



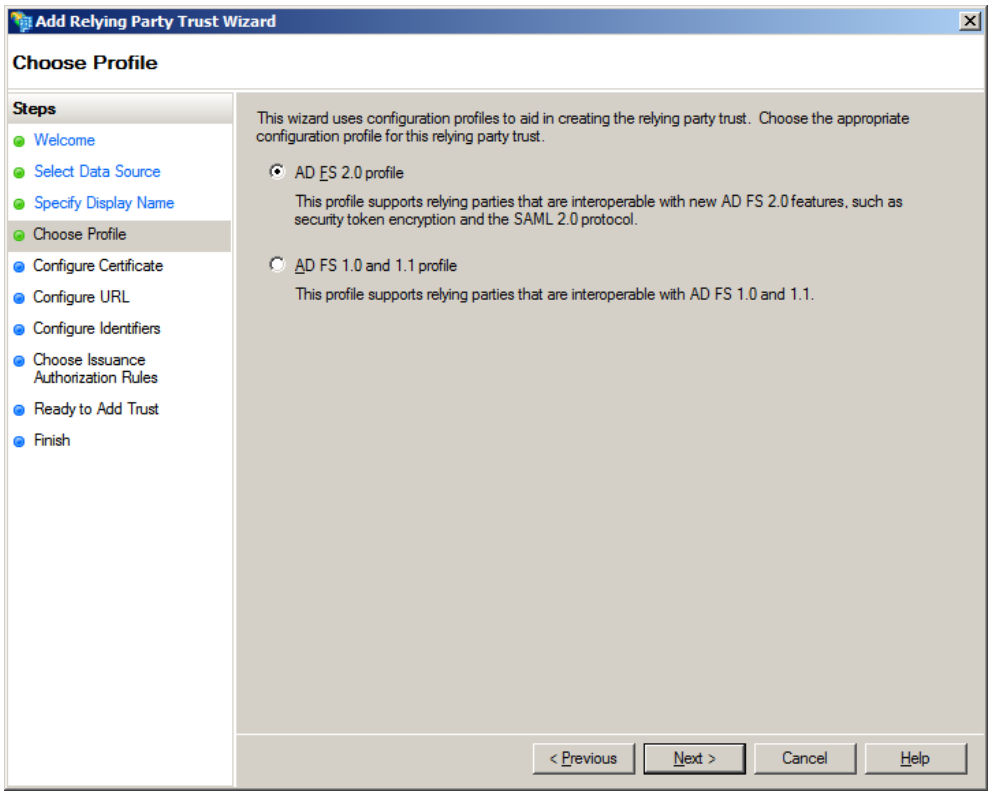
Kies 'Manually' zodat je alle configuratie parameters zelf moet invullen:



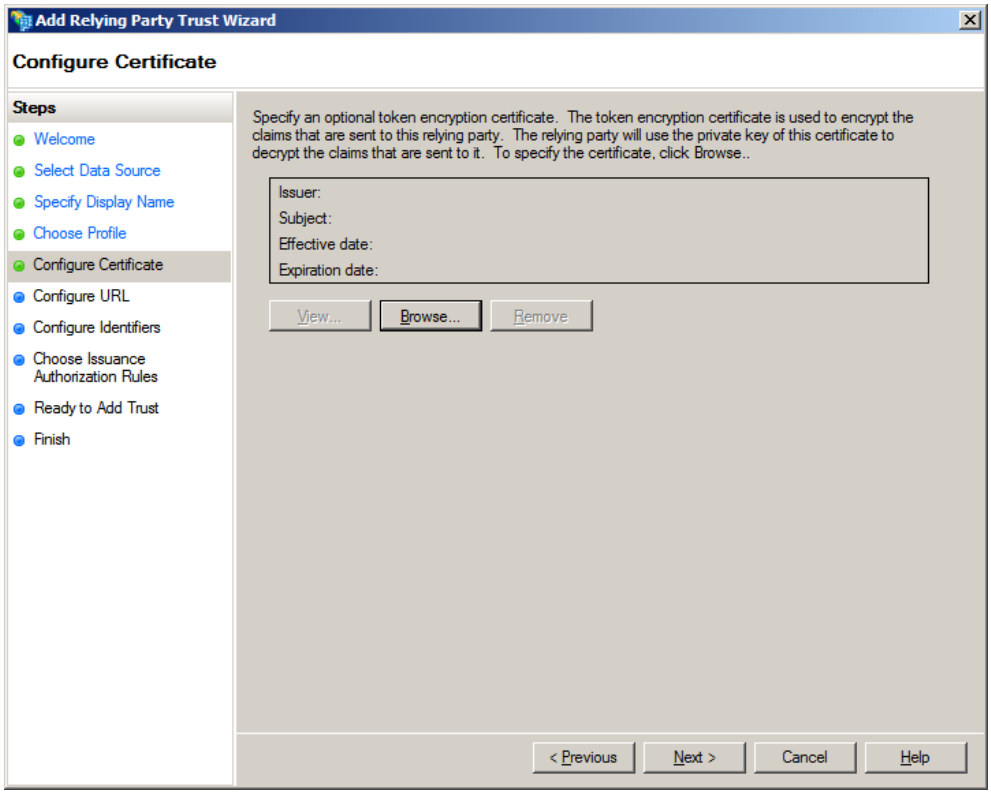
Tik een vrij te kiezen naam in voor deze trust party:



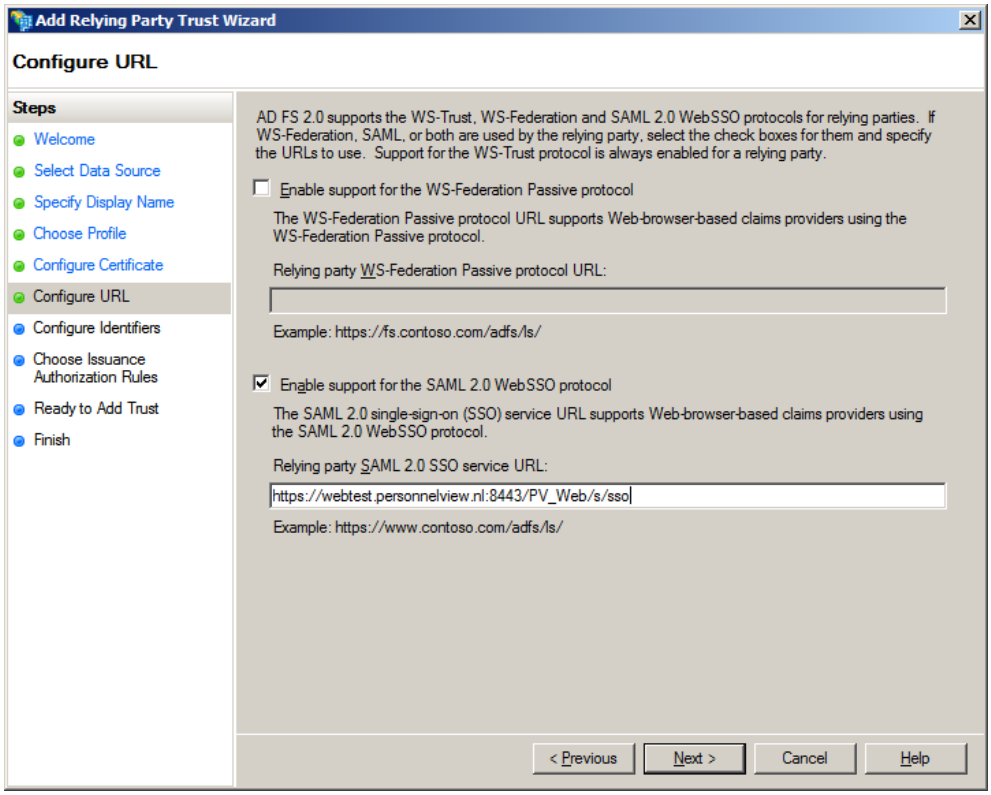
We gebruiken het ADFS 2.0 profile:



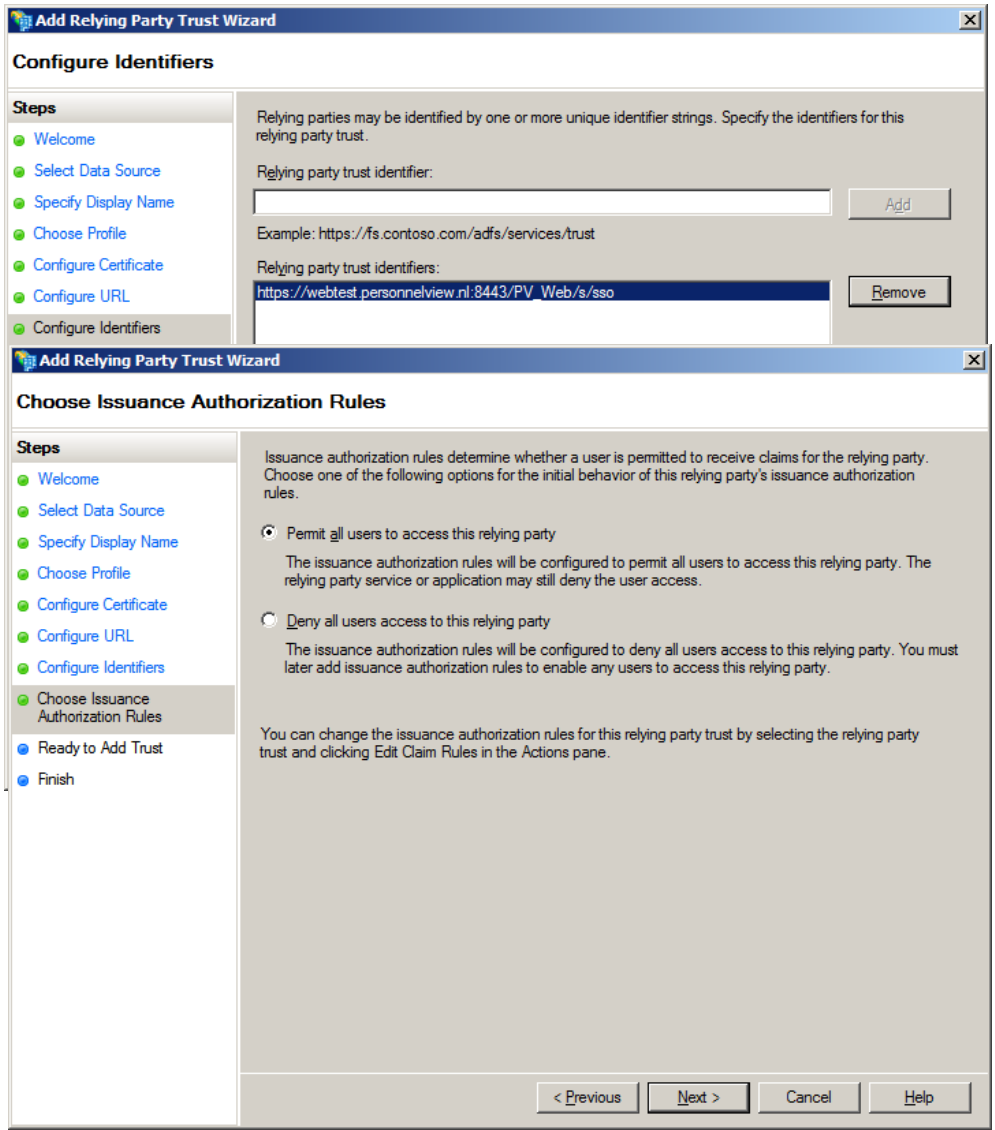
We gebruiken geen encryptie certificate, dus klik op next:



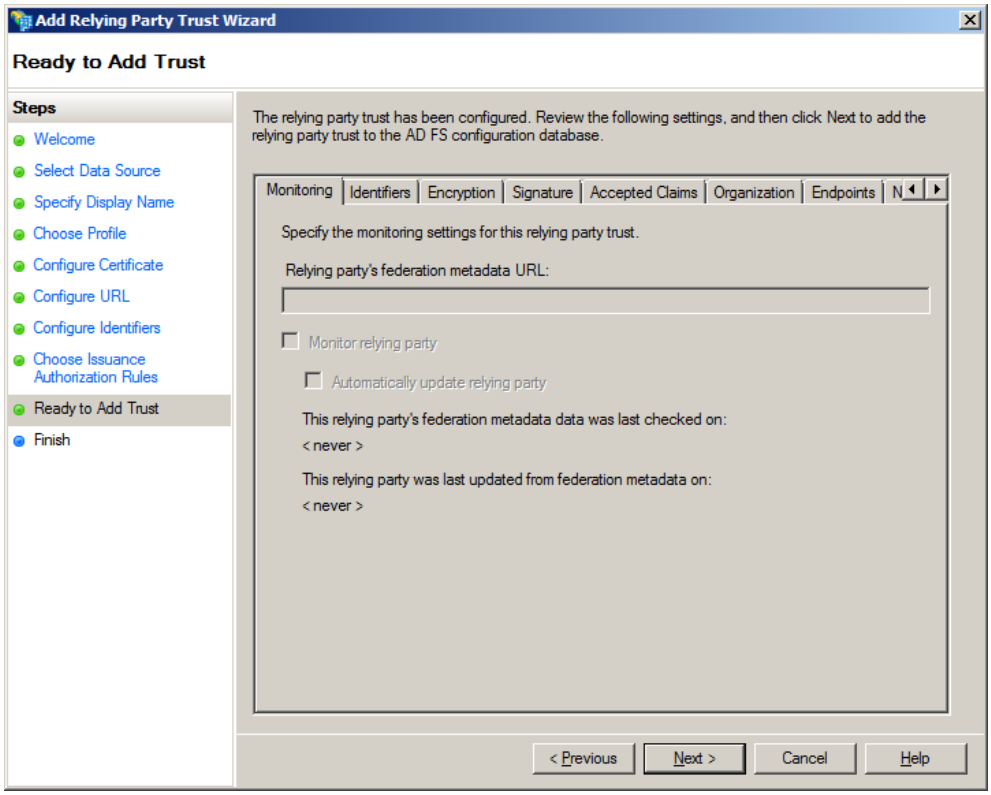
Type nu de URL in waaronder de Composer server SSO servlet te benaderen is:



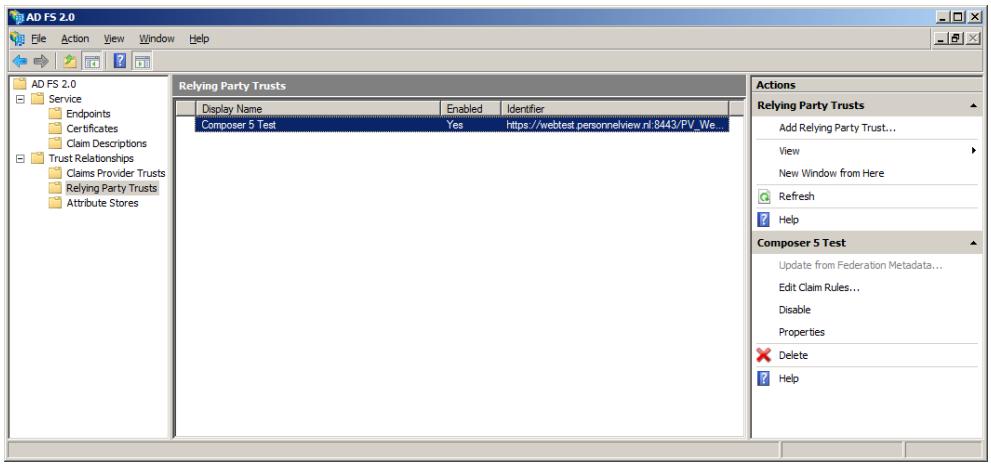
De relying trust party:



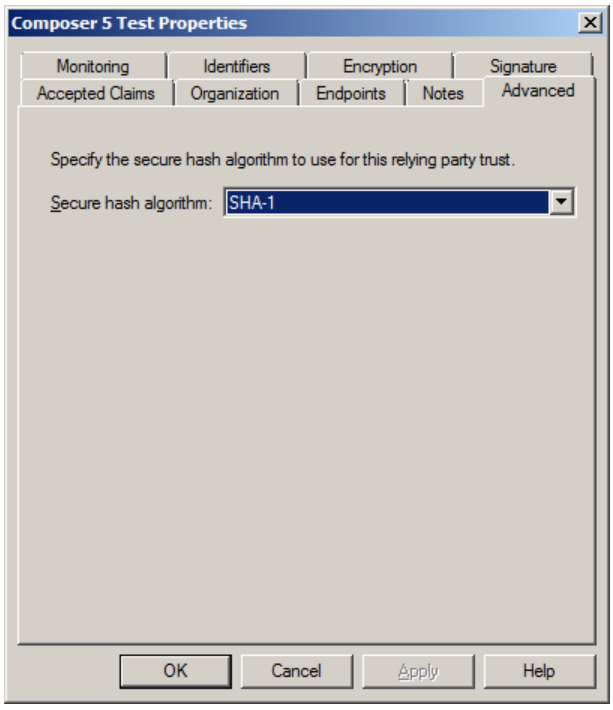
Permit all users en Click op Next



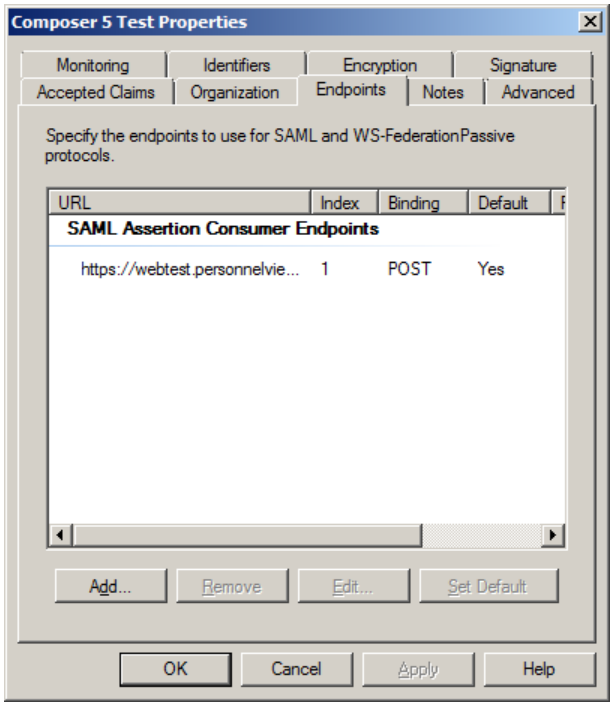
De wizard is nu afgerond.



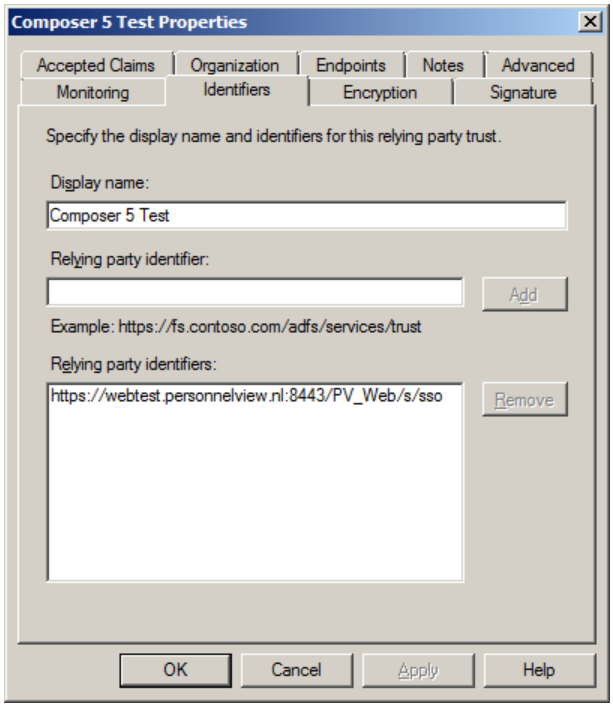
Wijzig nu (double click op party trust) de net aangemaakte trust party naar SHA-1 hashing (Composer SSO library ondersteunt alleen SHA-1):



Controlleer of je een POST endpoint hebt:

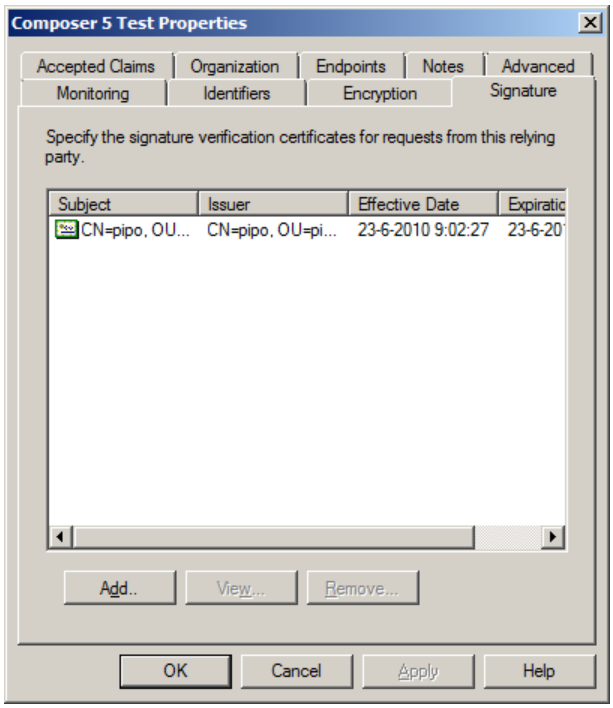


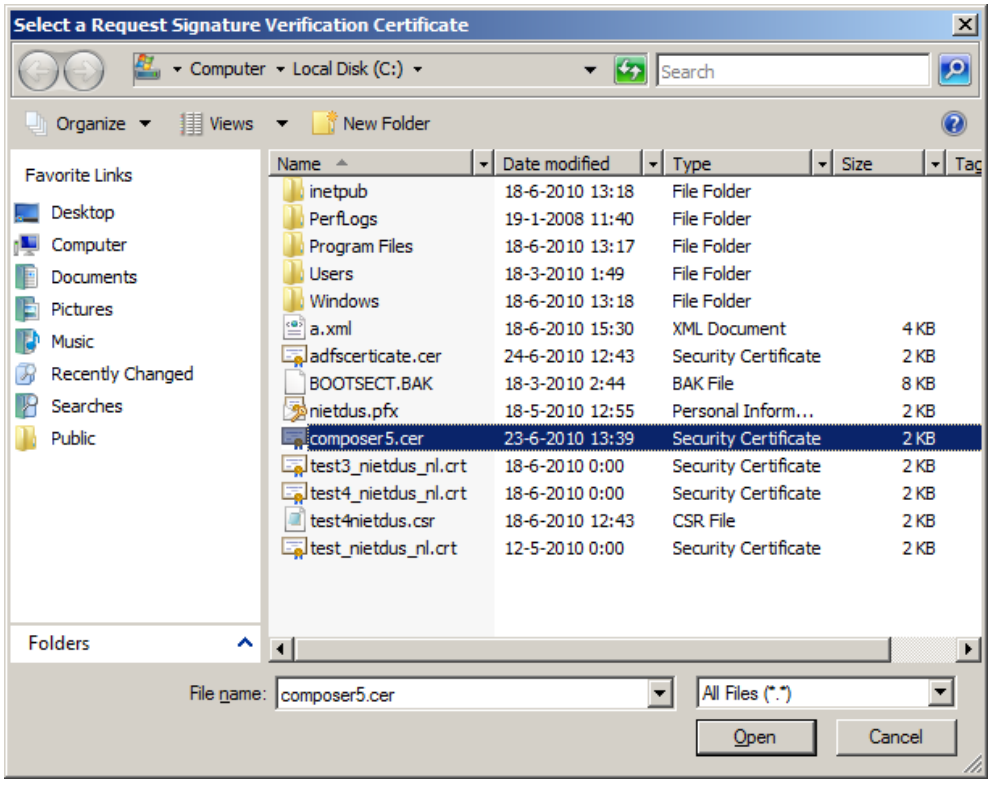
en of de relying party identifiers goed staan:



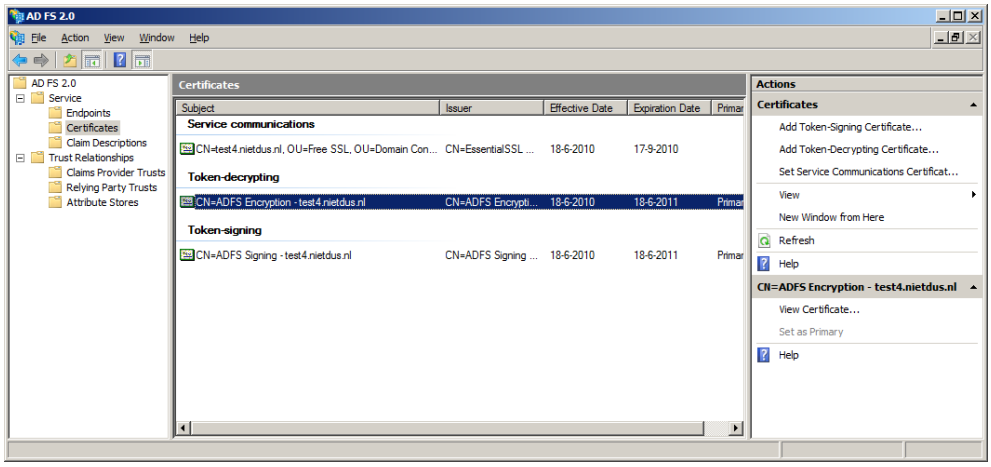
Voeg het certificate (met portecle zelf aangemaakt en gexporteerd certificate) van de Composer server (in de composer5_keystore.jks) toe door de .pem file in te lezen in het 'signature tab' van ADFSv2.

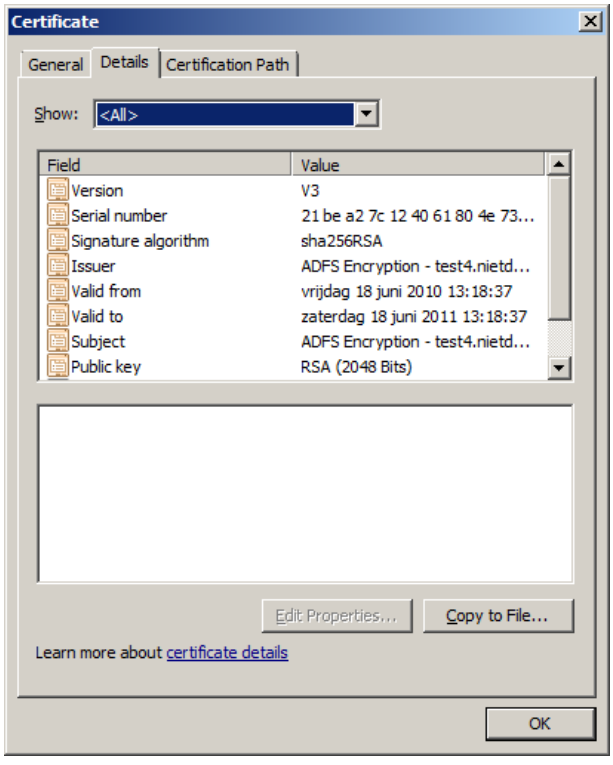
De key wordt door Composer gebruikt om alle requests naar de IDP te signen. Met het certificate behorende bij deze key kan de ADFS server controlleren of de requests wel bij Composer vandaan komen.



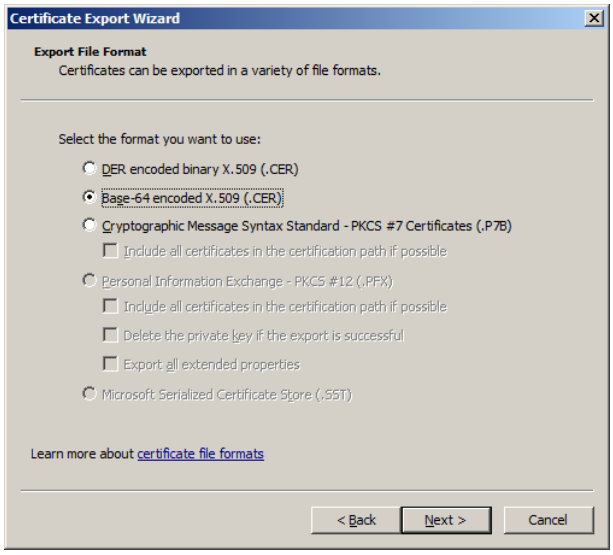


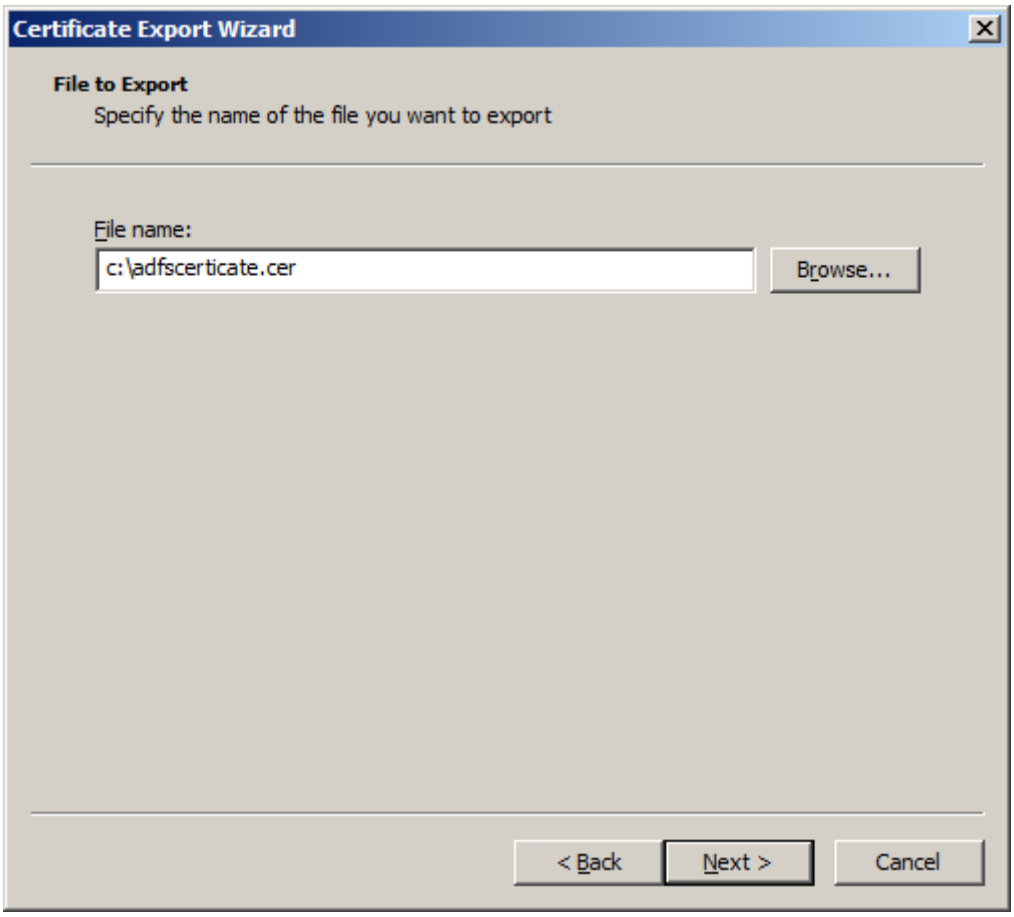
Exporteer nu het signing certificate van de ADFSv2 server om in te lezen in Composer. Dit certificate gebruikt de ADFS server om alle antwoorden op de requests van de Composer SP (service provider) te signen. Met het certificate kan Composer controlleren of het antwoord wel echt van de goede IDP vandaan komt en of het geldig en ongewijzigd is.





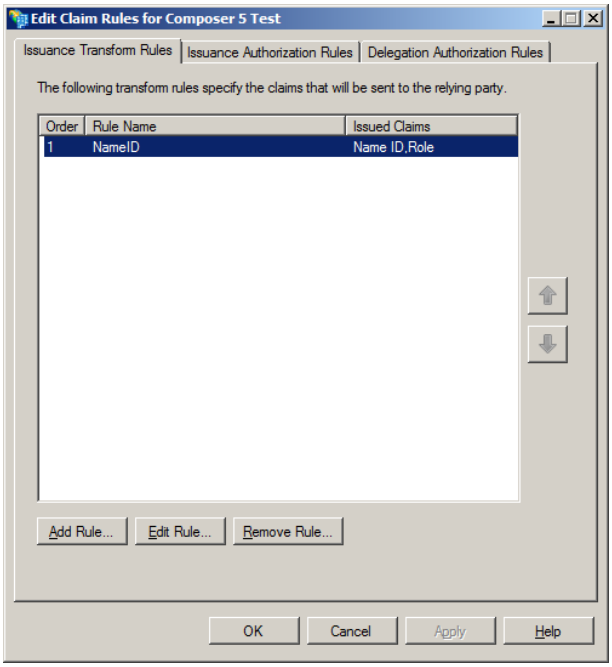
Copy to file:



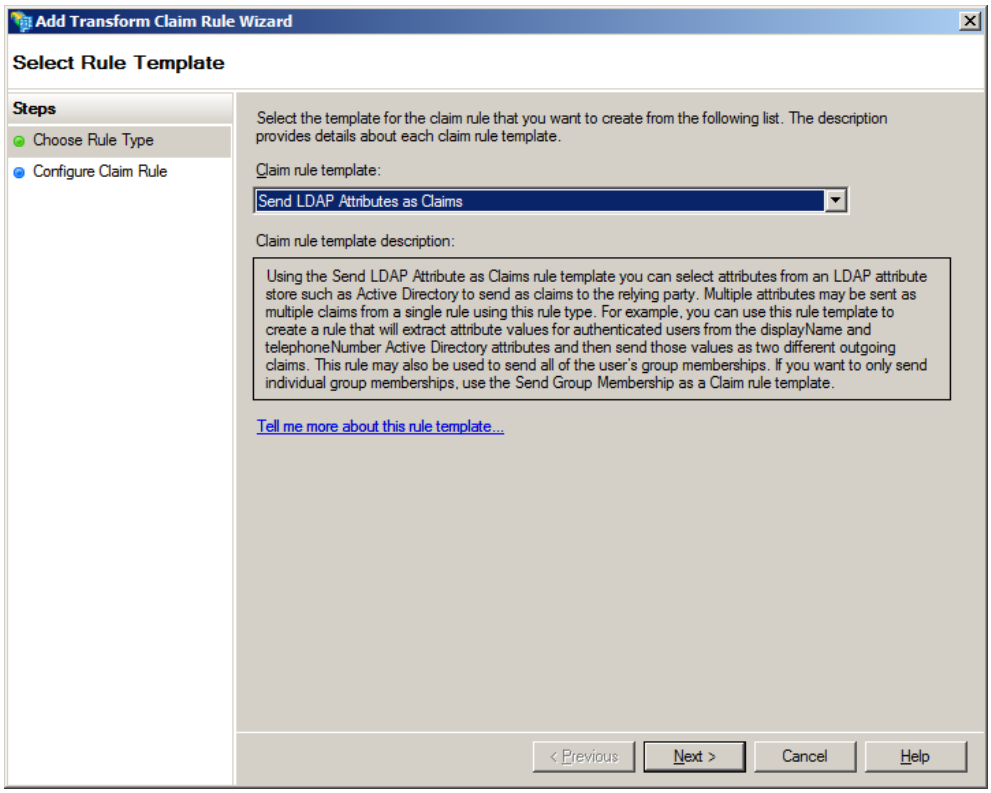


Copieer dit certificate naar de Composer server en lees het in de composer5_keystore.jks met de portecle tool!

Configureer nu de claim definitie welke terug geven moet worden op een identity request;

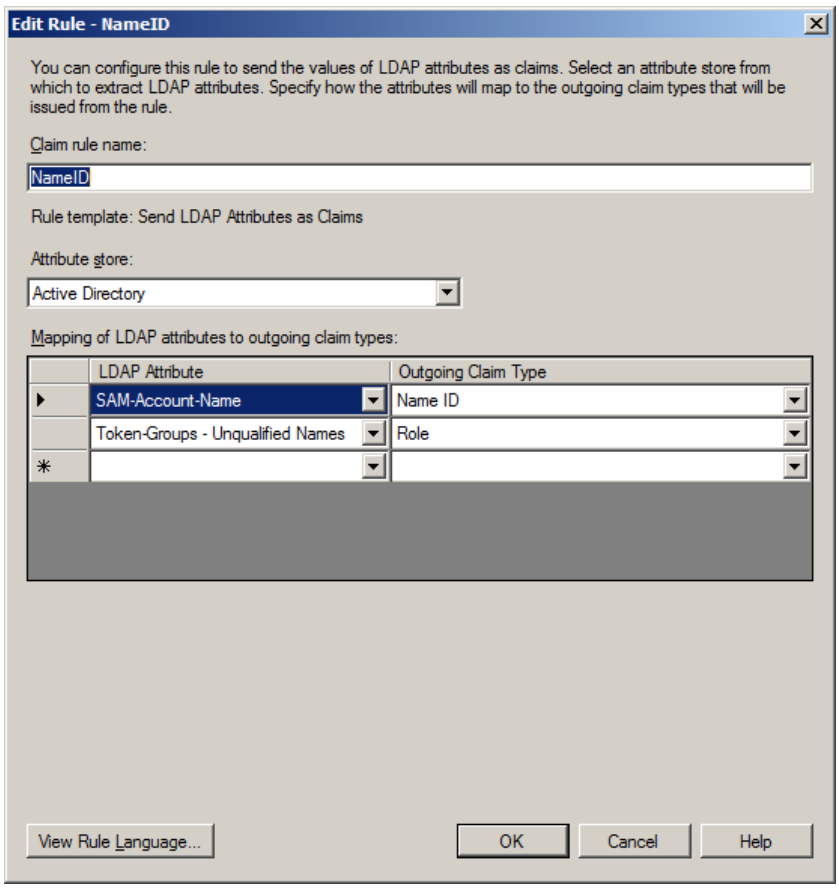


Click op 'Add Rule'



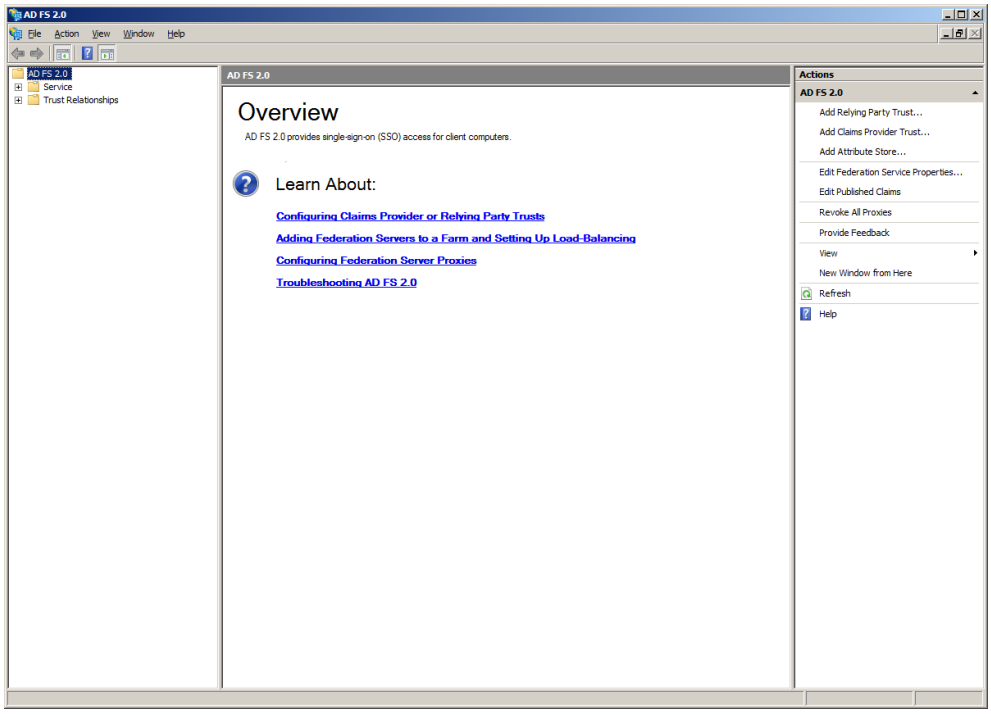
Kies voor 'Send LDAP Attributes as Claims' en dan Next>

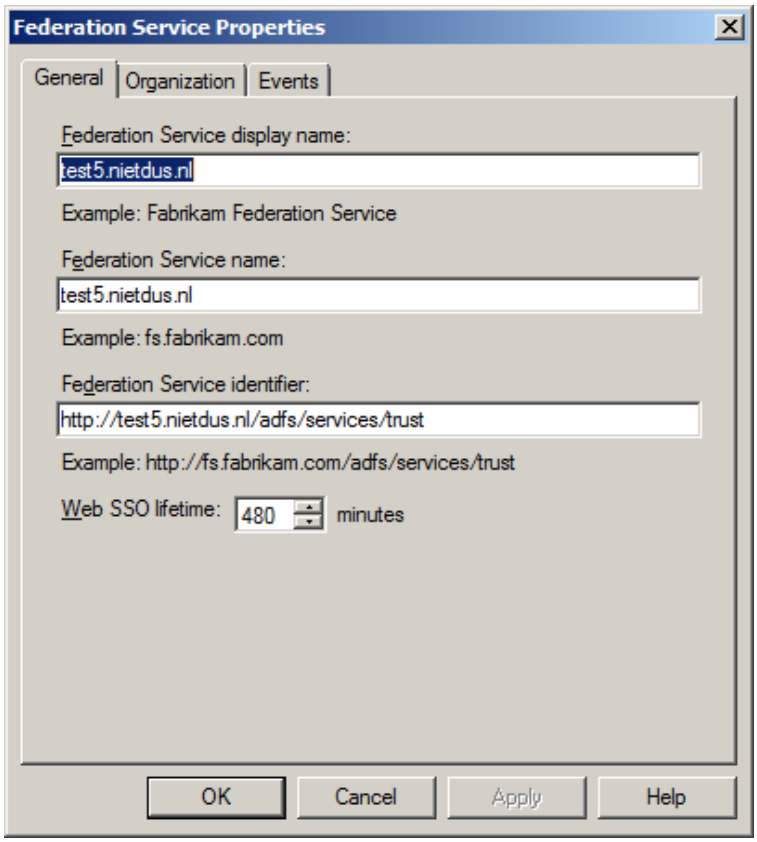
Configureer nu een twee claim attributen met Active Directory as attribute store. Deze definitie stuurt het windows inlog account en alle rol namen vanuit het AD door in de claim. Een claim is een soort bewijs welke door het ADFSv2 (in de rol als identity provider) wordt uitgegeven dat de gebruiker is wie hij zegt dat hij is. Deze claim wordt ge-signed met het eerder geconfigureerde certificate. Alle attributen binnen een claim kun je configureren. Composer heeft slechts twee SAML attributen nodig; NameID en Role. Het attribuut NameID moet een geldig username binnen Composer bevatten en de rollen lijst moet een rol met de naam 'composer' bevatten.



De endpoint naam van de ADFS server kun je als volgt veranderen als dat nodig is. Dit is de url waarop de adfsv2 server van buiten af benaderbaar is en welke ook door de adfsv2 server gecontrolleerd wordt (staat in de initiele AuthRequest van Composer);

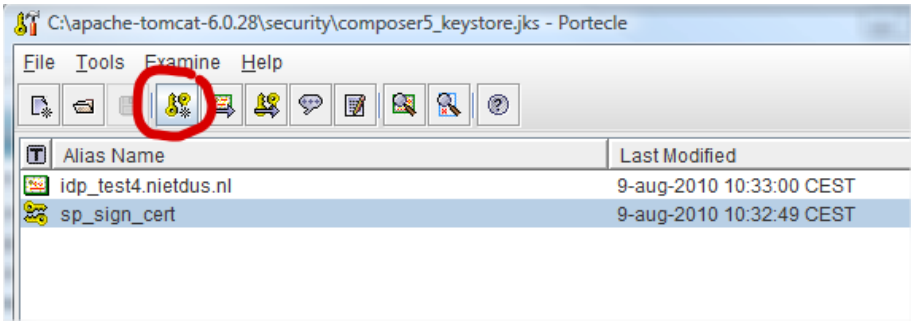
Rechtermuis toets op "ADFS V2" en dan 'Edit Federation Service Properties'



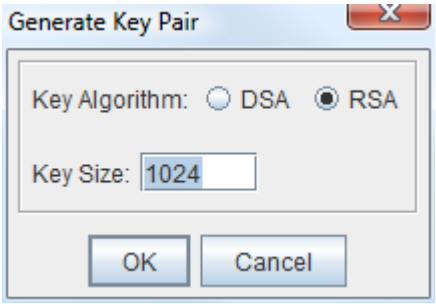


Self Sign Certificate aanmaken binnen Composer

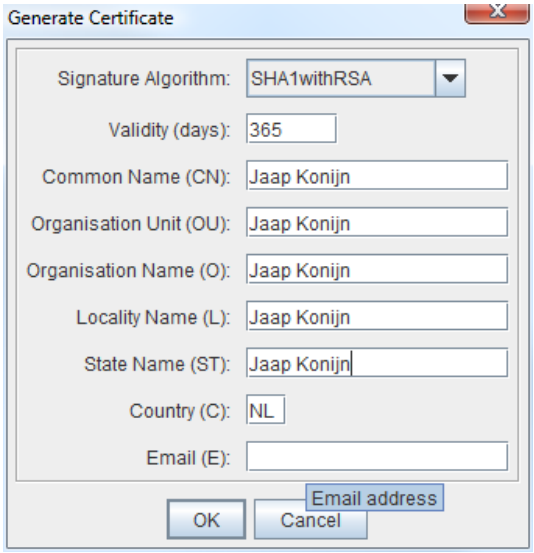
Aanmaken van een self signed certificate en exporten van sign key/certificate voor de SAML2 requests van Composer 5. Gebruik hiervoor de tool portecle-1.5 :



Kies voor 1024 RSA:

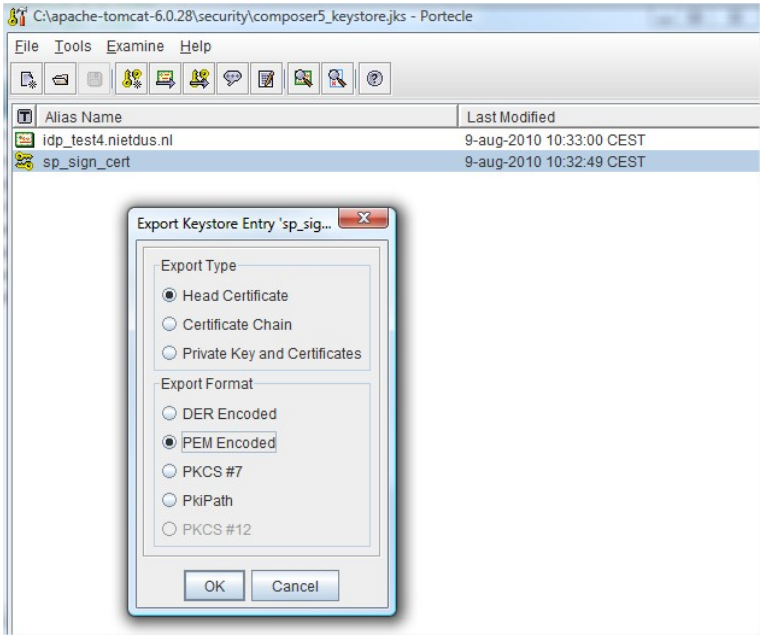


Vul gegevens van het certificate in en Click OK:

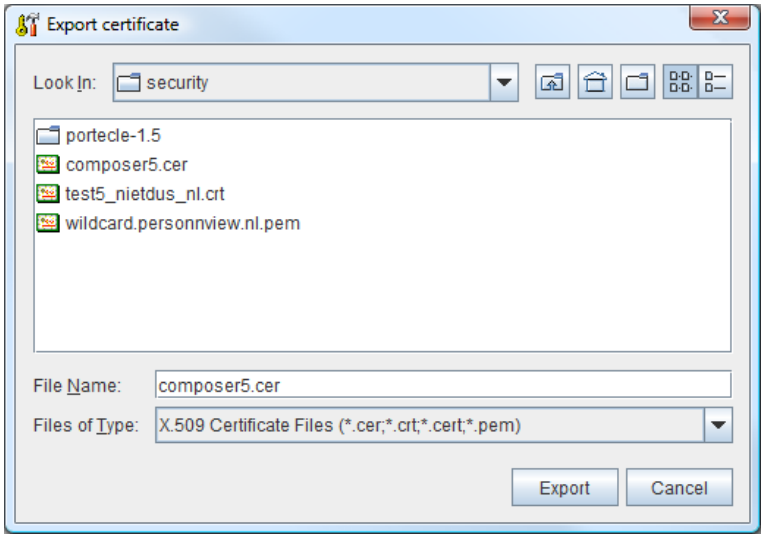


Geef dit key pair de naam 'sp_sign_cert'

Klik nu rechtermuis toets en kies voor export en stel export in op 'Head Certificate' en 'PEM Encoded':



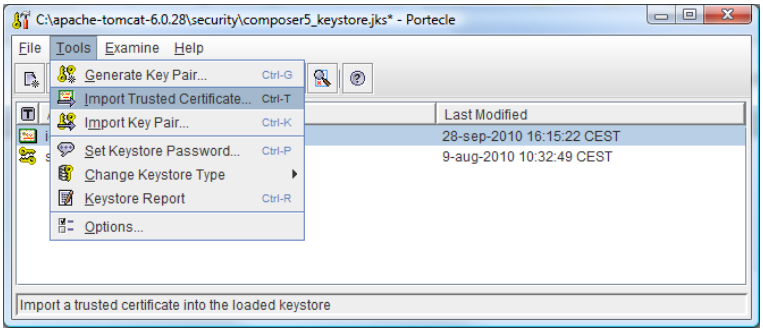
En geef de file de naam composer5.cer:



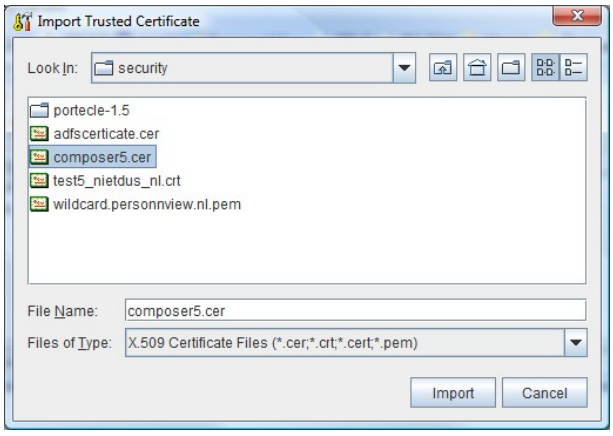
Je moet dit certificate bewaren en copieren naar de ADFSv2 server om hem daar tijdens de configuratie van de trust party in te lezen.

Inlezen ADFSv2 certificate binnen Composer

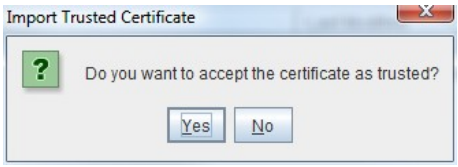
De certificaten van Composer staan in een keystore met de filenaam apache-tomcat-*/security/composer5_keystore.jks . Dit is het certificate wat ADFSv2 gebruikt om het antwoord te signen, en wat Composer 5 moet controlleren. Gebruik hiervoor de tool portecle-1.5 :



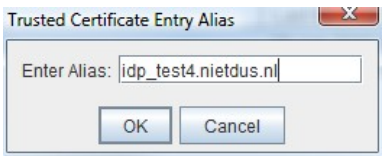
Kies de composer5.cer file welke je tijdens het configureren van de ADFSv2 server hebt aangemaakt:



Je kunt 1 of meer meldingen krijgen over dat de trust party niet valt te controlleren. Dit hoort:



Geef het certificate de naam van het domein van de ADFSv2 server bv.:



en Klik op OK