Kennisdeling Mutual Certificate opzet
in .NET

# **Doel van dit document**

De opzet van mutual SSL om VLOK als bron op het Magda platform aan te sluiten verliep niet rimpelloos. VLOK is geschreven in .NET technologie, het Magda platform werkt met Oracle/Java en deze 2 technologie-stacks gaan op een andere manier om met certificaten.

Onderstaande documentatie is gemaakt als kennisdeling zodat andere bronnen die via .NET werken kunnen bekijken op welke manier het VLOK project met succes is ontsloten via Mutual Certificate.

# **Oplossing**

## Certificaten

### Certificaten server

* Server certificaat (Private key)
* Client message certificaat (Public key)

### Certificaten client

* Server certificaat (Public key)
* Client message certificaat (Private key)

## Server WCF opzet

### Binding

Voor de binding maken we een customBinding aan met authenticationMode MutualCertificate. (Voor pure Mutual SSL moet je SecureConnection nemen, maar dit wordt niet beschreven in deze documentatie. We zijn overeengekomen dat MutualCertificate volstaat)

<binding name="SecureMutualSslCustomBinding">

 <textMessageEncoding messageVersion="Soap11" />

<security authenticationMode="MutualCertificate" allowInsecureTransport="false" enableUnsecuredResponse="true" messageProtectionOrder="EncryptBeforeSign" defaultAlgorithmSuite="Basic128" securityHeaderLayout="Lax" includeTimestamp="false" messageSecurityVersion="WSSecurity10WSTrustFebruary2005WSSecureConversationFebruary2005WSSecurityPolicy11BasicSecurityProfile10" requireSignatureConfirmation="false" canRenewSecurityContextToken="true">

 <issuedTokenParameters useStrTransform="true" />

 <localServiceSettings detectReplays="false" />

<secureConversationBootstrap defaultAlgorithmSuite="Default" authenticationMode="MutualSslNegotiated" requireDerivedKeys="true" includeTimestamp="true" messageSecurityVersion="WSSecurity11WSTrustFebruary2005WSSecureConversationFebruary2005WSSecurityPolicy11BasicSecurityProfile10" requireSignatureConfirmation="false">

 <localClientSettings detectReplays="true" />

 <localServiceSettings detectReplays="true" />

 </secureConversationBootstrap>

 </security>

 <httpsTransport requireClientCertificate="true" maxReceivedMessageSize="2147483647" />

</binding>

### Behavior

Hier stellen we het server certificaat in en de manier waarop het client message certificaat gevalideerd moet worden.

<behavior name="MutualSslBehavior">

 <serviceCredentials>

 <clientCertificate>

<authentication certificateValidationMode="PeerOrChainTrust" revocationMode="NoCheck" trustedStoreLocation="LocalMachine" />

 </clientCertificate>

<serviceCertificate findValue="**Server certificaat**" storeLocation="LocalMachine" storeName="My" x509FindType="FindBySubjectName" />

 </serviceCredentials>

</behavior>

Het server certificaat moet je hier zelf nog aanvullen.

### Service endpoint

 <service name="**Naam service**" behaviorConfiguration="MutualSslBehavior">

<endpoint address="**address**" binding="customBinding" bindingConfiguration="SecureMutualSslCustomBinding" contract="**contract**" bindingNamespace="**namespace**"></endpoint>

</service>

Hier moet je nog een aantal velden aanvullen met je eigen gegevens.

## Client Soap UI opzet

### Opstarten

We maken een nieuw SOAP UI project aan. We geven het een project naam en vullen de WSDL link in. Er wordt een nieuw project gecreerd met het juiste formaat.

### Security configuratie

We openen de configuratie van de map die we hebben aangemaakt. Onder de tab WS-Security Configurations gaan we eerst de keystore toevoegen. Hier moeten we ook het paswoord van de keystore invullen.

Daarna maken we een Outgoing configuration aan. Hier in voegen we 2 entries toe. Encryption en Signature. Encryption moet boven signature staan.

#### Encryption

We selecteren de keystore. Als Alias nemen we het publieke server certificaat en vullen het Password aan. Als Key Identifier Type selecteren we Issuer Name and Serial Number. We vinken dat hij een Encrypted key moet maken.



#### Signature

Hier selecteren we terug de keystore maar nemen als Alias het client message certificaat. We vullen ook het password aan. Als Key Identifier Type selecteren we Binary Security Token en we duiden aan dat hij maar 1 certificaat moet gebruiken voor het signen.



### Authenticatie

Op de request voegen we basic authenticatie toe.



Bij Authorization selecteren we Add New Authorization en voegen toe van het type Basic.



We selecteren bij Outgoing WSS de configuratie die we hebben aangemaakt. Als er op de server met een bepaald account moet worden ingelogd vul je de Username en Password nog aan.