Consumer Guide SecuredEchoService 1.0 & 2.0

Consumer Guide SecuredEchoService 1.0 & 2.0.pdf

Introduction

This guide explains how to consume a FSB-service with SOAP UI (SOAP UI is a open-source application for testing webservices).

Before developing the SOAP-proxy in your application, we recommend to use SOAP UI first. When all goes well with SOAP UI, then you are sure that the connectivity-, authentication- and authorisation-prerequisites on your environment are ok.

Next to the EchoServices (used to test the connectivity of your platform with the FSB-platform), the SecuredEchoServices are introduced to test the 2nd line of security : the signature of the SOAP request.

⚠ The security policies applied to the SecuredEchoServices are also valid for "real world" FSB services.

For those already experienced in SOAP UI, you can find the SOAP UI projects for the SecuredEchoServices in the Downloads section. The only change that needs to be done afterwards is to add your own keystore into the project (this is explained in sections Configure SOAP UI SecuredEchoService 1.0 or Configure SOAP UI SecuredEchoService 2.0).

There is also section to create a SOAP UI project from scratch. The WSDLs can be found in the Technical Documentation section.

Prerequisites

Before using one of these services, you should already have a FSB-specific certificate that is known and uploaded into the LDAP of BOSA.

You also need a keystore that contains the private key of your certificate. This keystore is needed to sign you requests with the private key in SOAP UI.

Remark: you also need this private key to be able to sign the message from a non SOAP UI application (in java, this will also be a keystore (JKS), in .net this will be the certificate store of your Windows server)

Downloads

SOAP UI projects

1.0

SecuredEchoService-soapui-project.xml

2.0

SecuredEchoService-with-FsbHeader-soapui-project.xml

Security Policies

There are two versions of SecuredEchoService :

- 1.0 : this version uses FSB policy 1 : the service only requires the signing of the body of the request. The response returned by FSB is not signed.
- 2.0 : this version uses FSB policy 2 : the service also requires the signing of the body of the request, but expects also a timestamp that must be signed as well. The response returned by FSB also contains a timestamp having body and timestamp signed.

⚠ A mixture of FSB services are either using policy 1 or policy 2. Which policy is used can be found in the FSB Service Catalogue http://dtservices.bosa.be/nl/services/fsb/web-services-families

SecuredEchoService Request/Response

1.0

This service echoes your message in tag <Echo>. The body of the message should be signed with your private key.

Request

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v1="http://fsb.belgium.be/EchoService/v1_00">
   <soapenv:Header/>
   <soapenv:Body>
      <v1:Echo>Hallo</v1:Echo>
   </soapenv:Body>
</soapenv:Envelope>
```

Example of a signed request (Done by SOAP UI)

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v1="http://fsb.belgium.be/EchoService/v1_00">
 <soapenv:Header>
  <wsse:Security
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wsse
curity-secext-1.0.xsd"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssec
```

```
urity-utility-1.0.xsd">
   <wsse:BinarySecurityToken
EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-so
ap-message-security-1.0#Base64Binary"
ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-
token-profile-1.0#X509v3"
wsu:Id="X509-C09327BBC081CD48AF1530620106521164">---PUBLIC
CERTIFICATE---</wsse:BinarySecurityToken>
   <ds:Signature Id="SIG-C09327BBC081CD48AF1530620106523168"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    <ds:SignedInfo>
     <ds:CanonicalizationMethod
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
      <ec:InclusiveNamespaces PrefixList="soapenv v1"
xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"/>
     </ds:CanonicalizationMethod>
     <ds:SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
     <ds:Reference URI="#id-C09327BBC081CD48AF1530620106521167">
      <ds:Transforms>
       <ds:Transform
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
        <ec:InclusiveNamespaces PrefixList="v1"
xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"/>
       </ds:Transform>
      </ds:Transforms>
      <ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>5LQd99xBdPjnKWJ8HY0d8tT+4UI=</ds:DigestValue>
     </ds:Reference>
    </ds:SignedInfo>

<ds:SignatureValue>NDJnNI40CCRRjGYrWvBgBBCVKC3e2uuwr1w16opRAyCLIZNTq+OIT
CixhUXEK8q2JUl7eXp9nJ5S...</ds:SignatureValue>
    <ds:KeyInfo Id="KI-C09327BBC081CD48AF1530620106521165">
     <wsse:SecurityTokenReference
wsu:Id="STR-C09327BBC081CD48AF1530620106521166">
      <wsse:Reference URI="#X509-C09327BBC081CD48AF1530620106521164"
ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-
token-profile-1.0#X509v3"/>
     </wsse:SecurityTokenReference>
    </ds:KeyInfo>
   </ds:Signature>
  </wsse:Security>
 </soapenv:Header>
 <soapenv:Body wsu:Id="id-C09327BBC081CD48AF1530620106521167"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssec
urity-utility-1.0.xsd">
```

```
  <v1:Echo>Hallo</v1:Echo>
 </soapenv:Body>
</soapenv:Envelope>
```

Remark: ---*PUBLIC CERTIFICATE*--- in tag BinarySecurityToken means that this part of the message will contain your public certificate in base64 format.

Response

```
<soapenv:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v1="http://fsb.belgium.be/EchoService/v1_00">
   <soapenv:Header/>
   <soapenv:Body wsu:Id="id-AF63E2D621E6DF22F4153062403726914"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssec
urity-utility-1.0.xsd">
      <v1:Echo>Hallo</v1:Echo>
   </soapenv:Body>
</soapenv:Envelope>
```

2.0

This service also echoes your message that is found in tag <Echo> of your request. The difference between SecuredEchoService v1 and SecuredEchoService v2 lies in the security specifications :

- Addtional information (messageId) is required in the header of the message (correlationId and processInstanceId are optional).
  - messageId should be an unique indentifier (the format is up to you). In case of a problem, FSB can investigate the logs/archives based on that messageId
  - correlationId is optional.This can f.e. be used to link several requests belonging to the same transaction
  - processInstanceId is also optional. This can f.e. be used to provide an unique identifier within the same correlationId.
- A timestamp is required in the header of the message
- the body and the message need to be signed
- the body and timestamp of the response are also signed (with a FSB-certificate cn=FEDERALSERVICEBUSINT or cn=FEDERALSERVICEBUSPROD)

Request

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:head="http://fsb.belgium.be/header"
xmlns:v2="http://fsb.belgium.be/EchoService/v2_00">
   <soapenv:Header>
      <head:fsbHeader>
         <head:messageId>je eigen nummering</head:messageId>
      </head:fsbHeader>
   </soapenv:Header>
   <soapenv:Body>
      <v2:Echo>Hallo</v2:Echo>
   </soapenv:Body>
</soapenv:Envelope>
```

Please note that tags <FsbHeader> and <messageId> requires namespace http://fsb.belgium.be/header.

Example of a signed request (Done by SOAP UI)

```
<soapenv:Envelope xmlns:head="http://fsb.belgium.be/header"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v2="http://fsb.belgium.be/EchoService/v2_00">
 <soapenv:Header>
  <wsse:Security
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wsse
curity-secext-1.0.xsd"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssec
urity-utility-1.0.xsd">
   <wsse:BinarySecurityToken
EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-so
ap-message-security-1.0#Base64Binary"
ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-
token-profile-1.0#X509v3"
wsu:Id="X509-C09327BBC081CD48AF1530619837283154">---PUBLIC
CERTIFICATE---</wsse:BinarySecurityToken>
   <ds:Signature Id="SIG-C09327BBC081CD48AF1530619837284158"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    <ds:SignedInfo>
     <ds:CanonicalizationMethod
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
      <ec:InclusiveNamespaces PrefixList="head soapenv v2"
xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"/>
     </ds:CanonicalizationMethod>
     <ds:SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
     <ds:Reference URI="#id-C09327BBC081CD48AF1530619837284157">
```

```
      <ds:Transforms>
       <ds:Transform
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
        <ec:InclusiveNamespaces PrefixList="head v2"
xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"/>
       </ds:Transform>
      </ds:Transforms>
      <ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>SMAYKfwfTEmzgsC9XdJDKjo8nKU=</ds:DigestValue>
     </ds:Reference>
     <ds:Reference URI="#TS-C09327BBC081CD48AF1530619837281153">
      <ds:Transforms>
       <ds:Transform
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
        <ec:InclusiveNamespaces PrefixList="wsse head soapenv v2"
xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"/>
       </ds:Transform>
      </ds:Transforms>
      <ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>YNz1Ux8b83EsKvLIKbnjH+flWWQ=</ds:DigestValue>
     </ds:Reference>
    </ds:SignedInfo>

<ds:SignatureValue>u8fJmkDtMWoT0tq7r6r489uOk3kaprS9/wG/brx5jv9cE3cMTzTTS
x9YFf5BrVzWMOt+oNAII4CG...=</ds:SignatureValue>
    <ds:KeyInfo Id="KI-C09327BBC081CD48AF1530619837283155">
     <wsse:SecurityTokenReference
wsu:Id="STR-C09327BBC081CD48AF1530619837283156">
      <wsse:Reference URI="#X509-C09327BBC081CD48AF1530619837283154"
ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-
token-profile-1.0#X509v3"/>
     </wsse:SecurityTokenReference>
    </ds:KeyInfo>
   </ds:Signature>
   <wsu:Timestamp wsu:Id="TS-C09327BBC081CD48AF1530619837281153">
    <wsu:Created>2018-07-03T12:10:37Z</wsu:Created>
    <wsu:Expires>2018-07-03T12:15:37Z</wsu:Expires>
   </wsu:Timestamp>
  </wsse:Security>
  <head:fsbHeader>
   <head:messageId>je eigen nummering</head:messageId>
  </head:fsbHeader>
 </soapenv:Header>
 <soapenv:Body wsu:Id="id-C09327BBC081CD48AF1530619837284157"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssec
urity-utility-1.0.xsd">
```

```
  <v2:Echo>Hallo</v2:Echo>
 </soapenv:Body>
</soapenv:Envelope>
```

Remarks:

- ---*PUBLIC CERTIFICATE*--- in tag BinarySecurityToken means that this part of the message will contain your public certificate in base64 format.
- Note the presence of a timestamp.

Example of a signed Response

```
<soapenv:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <env:Header xmlns:head="http://fsb.belgium.be/header"
xmlns:v2="http://fsb.belgium.be/EchoService/v2_00">
      <head:fsbHeader>
         <fsb:fsbTransactionId
xmlns:fsb="http://fsb.belgium.be/header">Wzt4R0JEMfnKXWC4x8NnvgAAAAM</fs
b:fsbTransactionId>
         <head:messageId>je eigen nummering</head:messageId>
      </head:fsbHeader>
      <wsse:Security soapenv:mustUnderstand="1"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wsse
curity-secext-1.0.xsd">
         <wsu:Timestamp wsu:Id="Timestamp-VPhrH1VV6MeJTPWter1tSA22"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssec
urity-utility-1.0.xsd">
            <wsu:Created>2018-07-03T13:21:12Z</wsu:Created>
            <wsu:Expires>2018-07-03T13:26:12Z</wsu:Expires>
         </wsu:Timestamp>
         <wsse:BinarySecurityToken
ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-
token-profile-1.0#X509v3"
EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-so
ap-message-security-1.0#Base64Binary"
wsu:Id="BST-uwgZ0lB5xMHB5TeShAo4Yg22"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssec
urity-utility-1.0.xsd">---PUBLIC CERTIFICATE OF
FSB---</wsse:BinarySecurityToken>
         <dsig:Signature
xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
            <dsig:SignedInfo>
               <dsig:CanonicalizationMethod
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
               <dsig:SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <dsig:Reference
URI="#Timestamp-VPhrH1VV6MeJTPWter1tSA22">
```

```
                  <dsig:Transforms>
                     <dsig:Transform
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                  </dsig:Transforms>
                  <dsig:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>

<dsig:DigestValue>AM+kyVH4HQzwzl+J+R2HB6cQCnE=</dsig:DigestValue>
               </dsig:Reference>
               <dsig:Reference
URI="#id-AF63E2D621E6DF22F4153062407238320">
                  <dsig:Transforms>
                     <dsig:Transform
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                  </dsig:Transforms>
                  <dsig:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>

<dsig:DigestValue>8YsBhb6fZ8jW+wtRrRNdhT35FVI=</dsig:DigestValue>
               </dsig:Reference>
            </dsig:SignedInfo>

<dsig:SignatureValue>vLaUjtaG3rPjqhggmHurzjXu1CF3NGhIi</dsig:SignatureVa
lue>
            <dsig:KeyInfo Id="KeyInfo-d418HG9w0b99BXbNsG41Vg22">
               <wsse:SecurityTokenReference>
                  <wsse:Reference URI="#BST-uwgZ0lB5xMHB5TeShAo4Yg22"
ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-
token-profile-1.0#X509v3"/>
               </wsse:SecurityTokenReference>
            </dsig:KeyInfo>
         </dsig:Signature>
      </wsse:Security>
   </env:Header>
   <env:Body wsu:Id="id-AF63E2D621E6DF22F4153062407238320"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssec
urity-utility-1.0.xsd" xmlns:head="http://fsb.belgium.be/header"
xmlns:v2="http://fsb.belgium.be/EchoService/v2_00">
```

```
        <v2:Echo>Echoing Hallo</v2:Echo>
    </env:Body>
</soapenv:Envelope>
```

Technical documentation (WSDL, XSD)

Service contracts (wsdl, xsd's) and endpoints are available via below links:

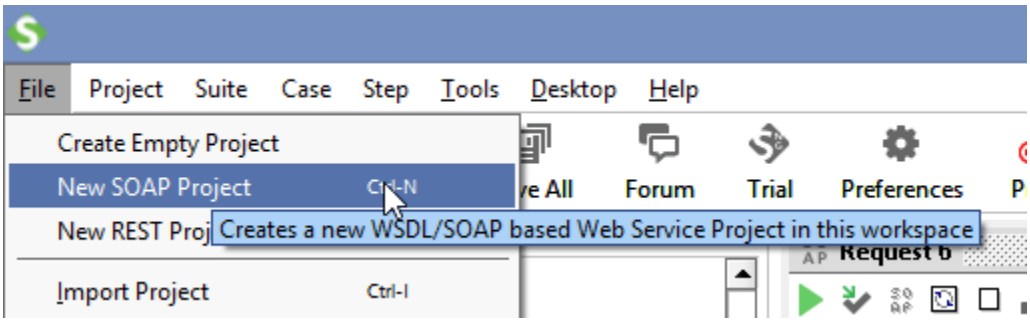
SecuredEchoService v1

SecuredEchoService v2

Create SOAP UI project SecuredEchoService 1.0

1° Open SOAP UI

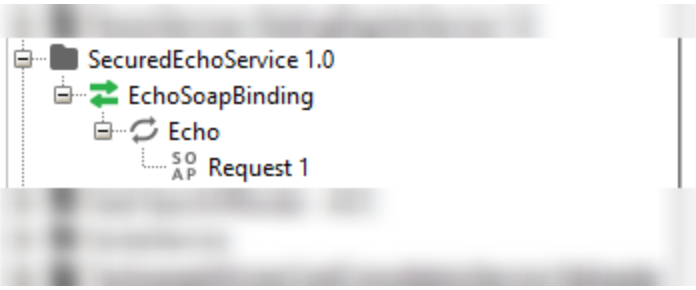
2° Select via Menu-File item "New SOAP Project"



3° Enter "https://fsb.services.int.belgium.be/1.00/CPS_SecuredEchoService?wsdl" to retrieve the WSDL (or you can download the WSDL from browser, and via button Browse... you can select the WSDL-file). Provide also a Project name.

⚠ When you can't get the WSDL from the URL, there is probably a firewall issue between your platform and FSB.

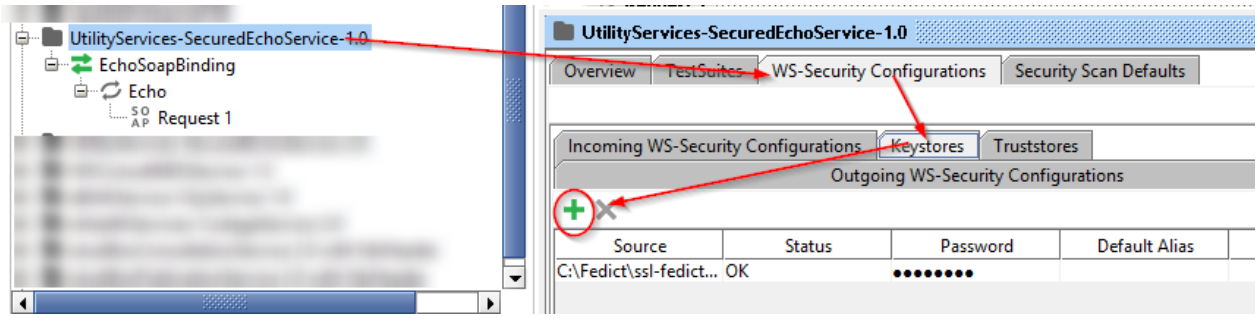
4° You'll see this in the navigator.



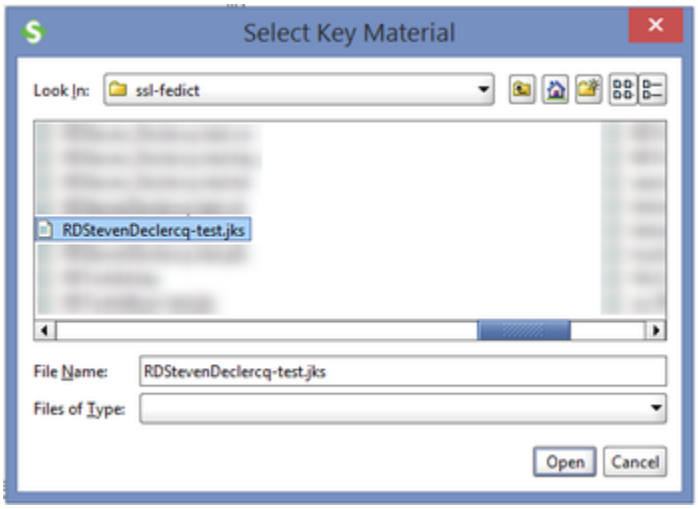
5° As the SecuredEchoService requires a signed request, the next sections explains how to add security

6° Add your own keystore.The keystore contains the private and public key of your certificate, which is required to sign the request.

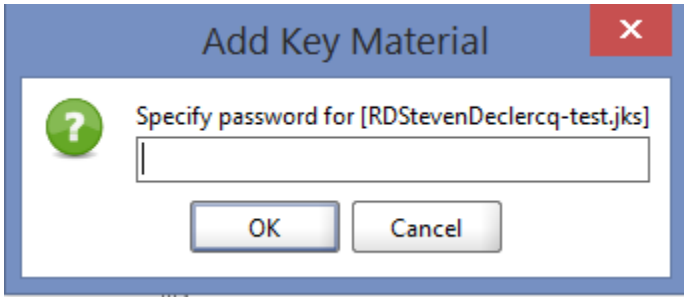
- Double click on the project. This will open the property screen. Go to WS Security Configuration/Keystores
- Click on the green '+' to add your keystore



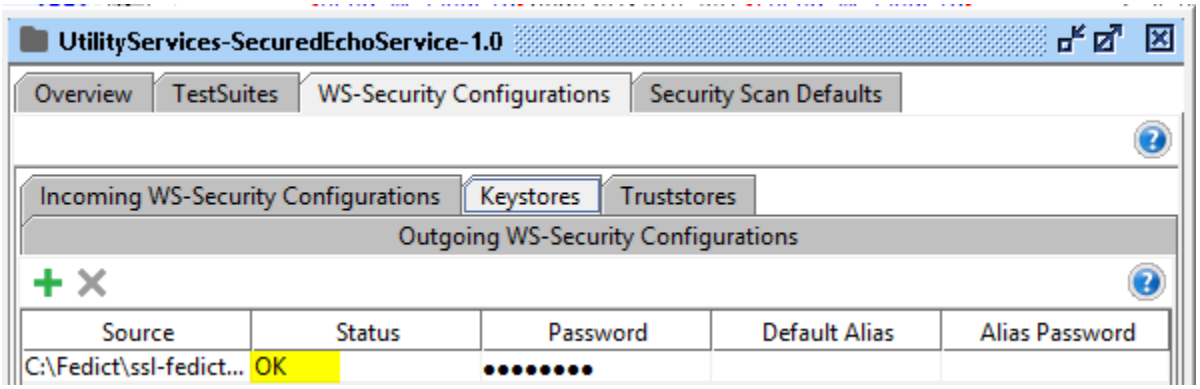
- Most probably, this will be a .jks file that contains the private and public key of your certificate



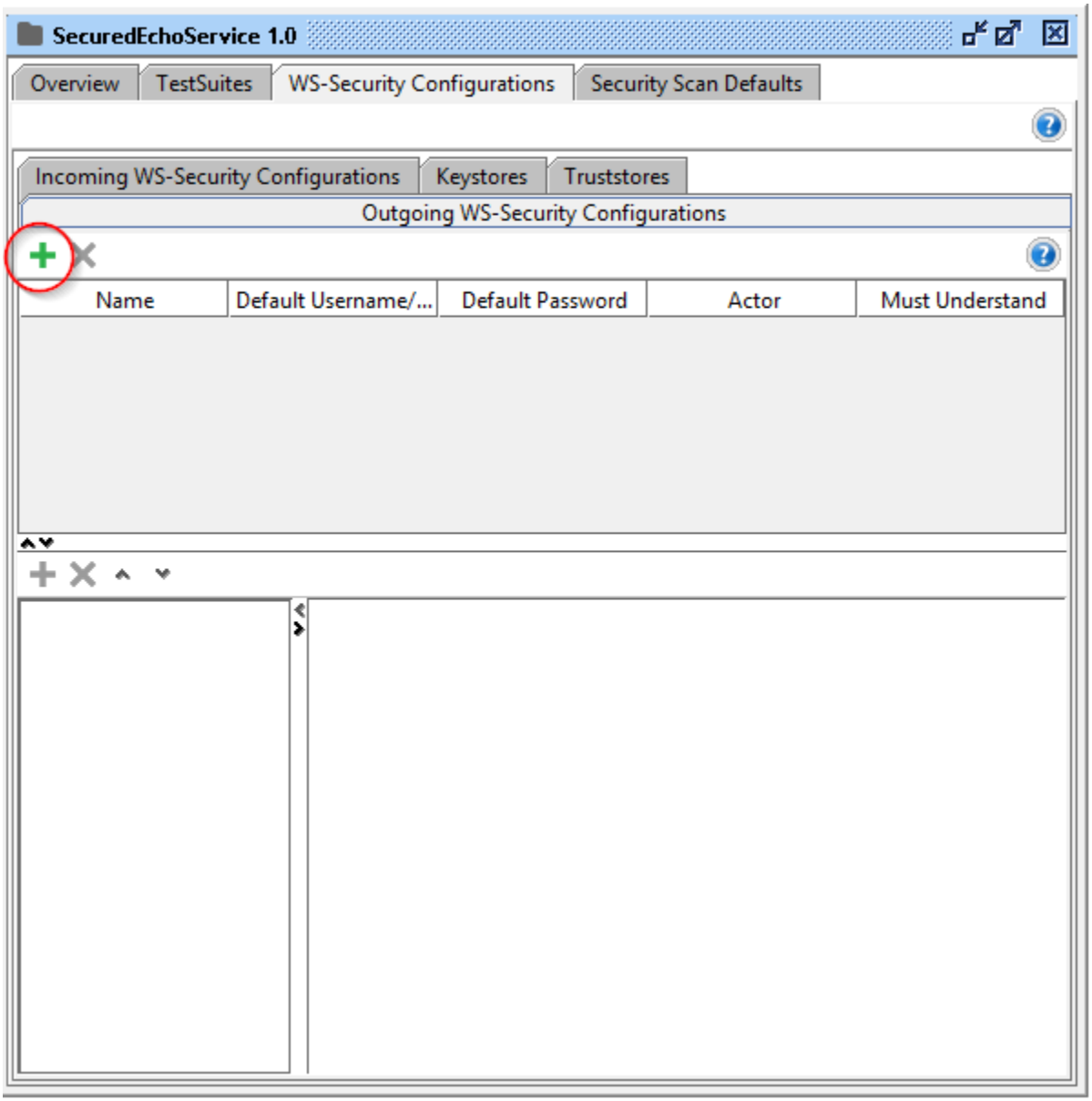
- Enter the password that you have assigned to your keystore



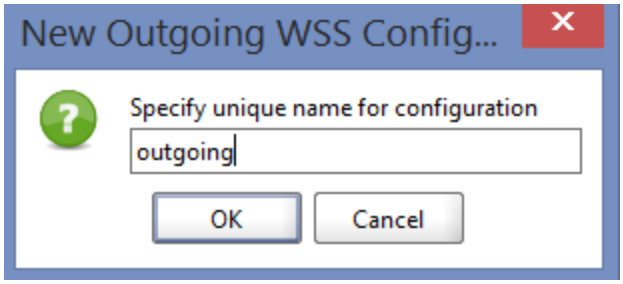
- When the entered password is correct, the status will be 'OK'



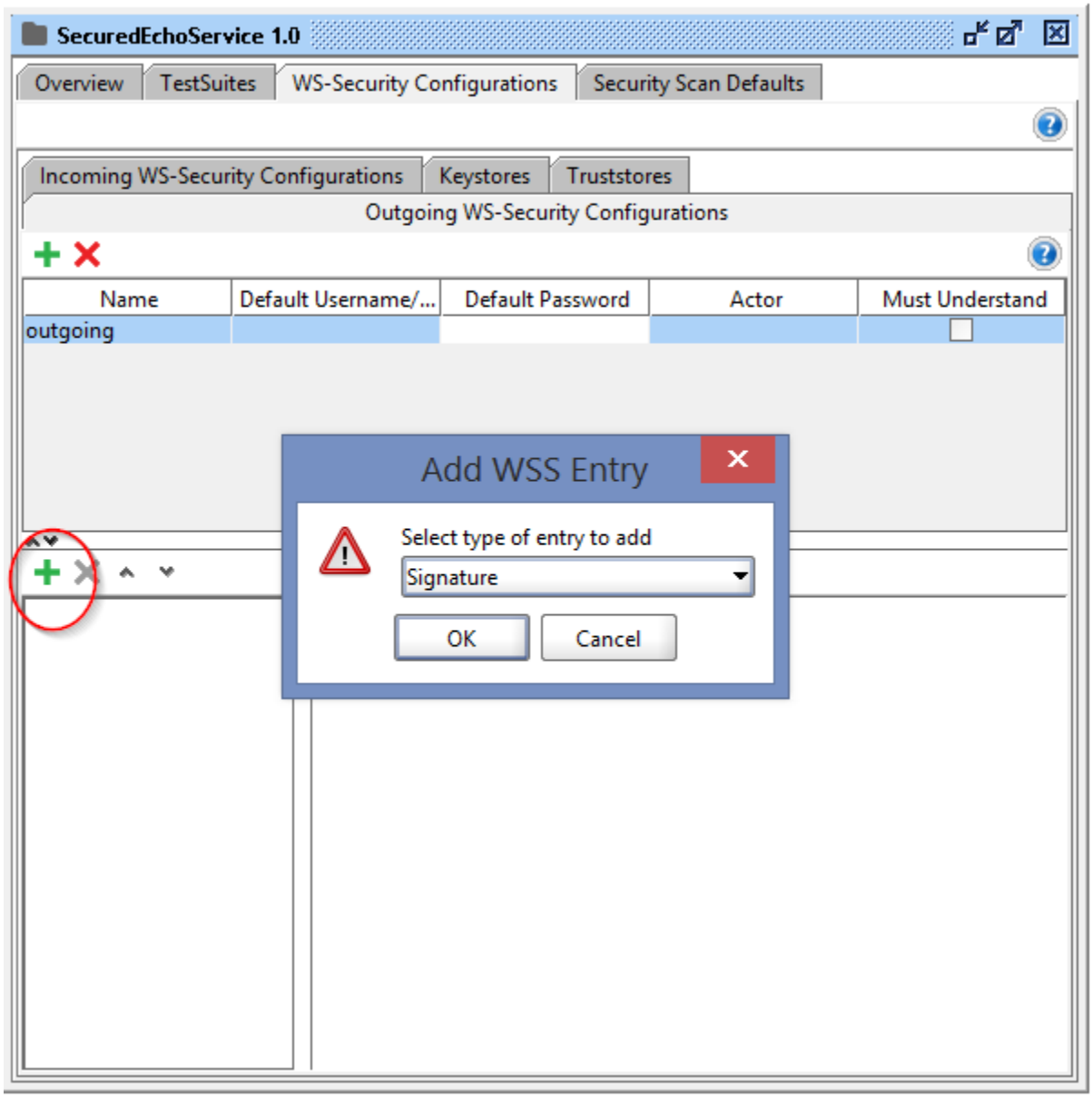
- Now we'll attach your keystore to the siging part of the request. Go to tabpage "Outgoing WS-Security Configurations", click on the green '+ '



- Give the Security Configuration a name (f.e. outgoing)



- Now click on the green '+' to select WSS entries, choose Signature



- provide the following information and close this window.

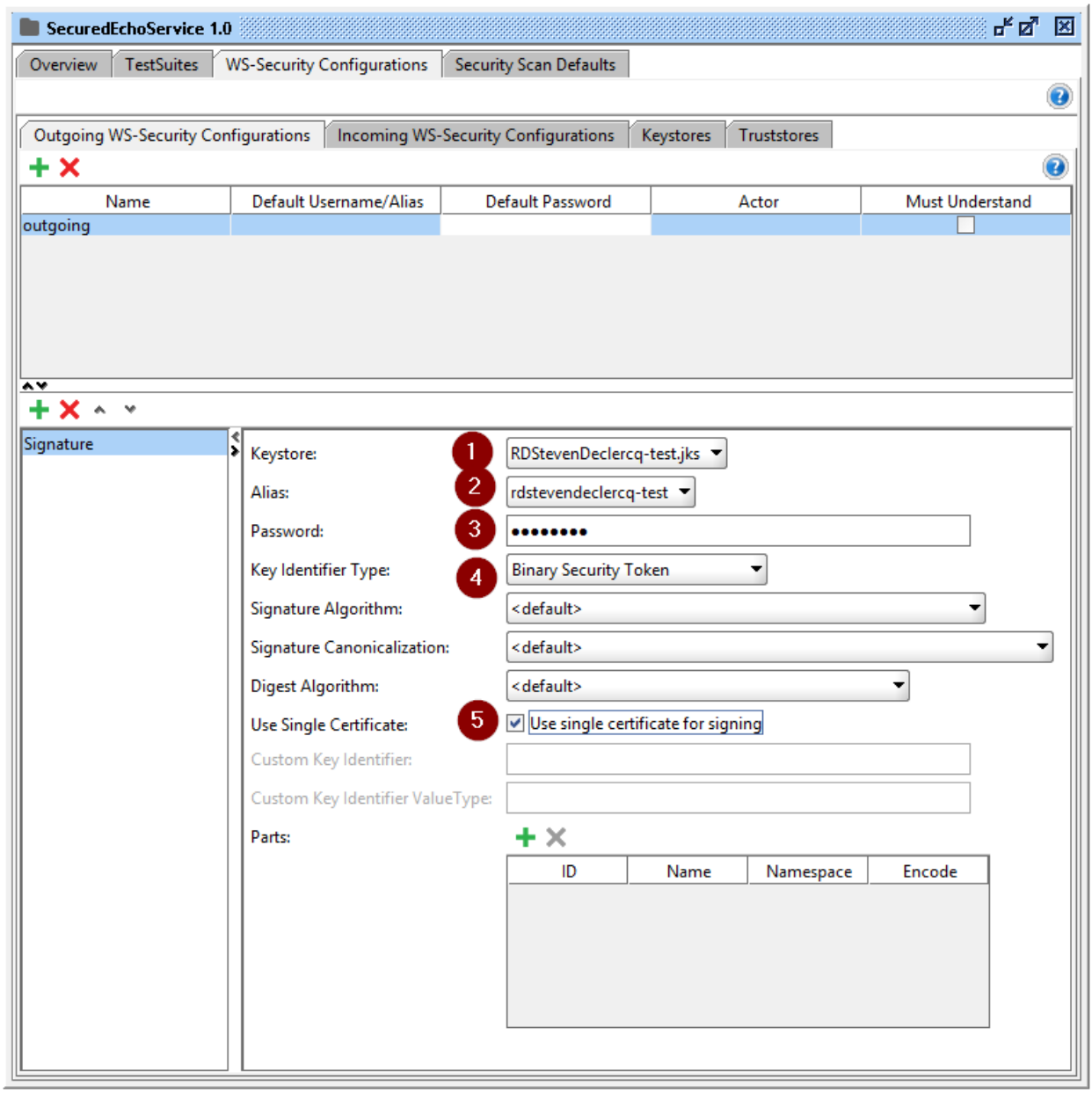
  1= filename of the keystore

  2= alias of private key

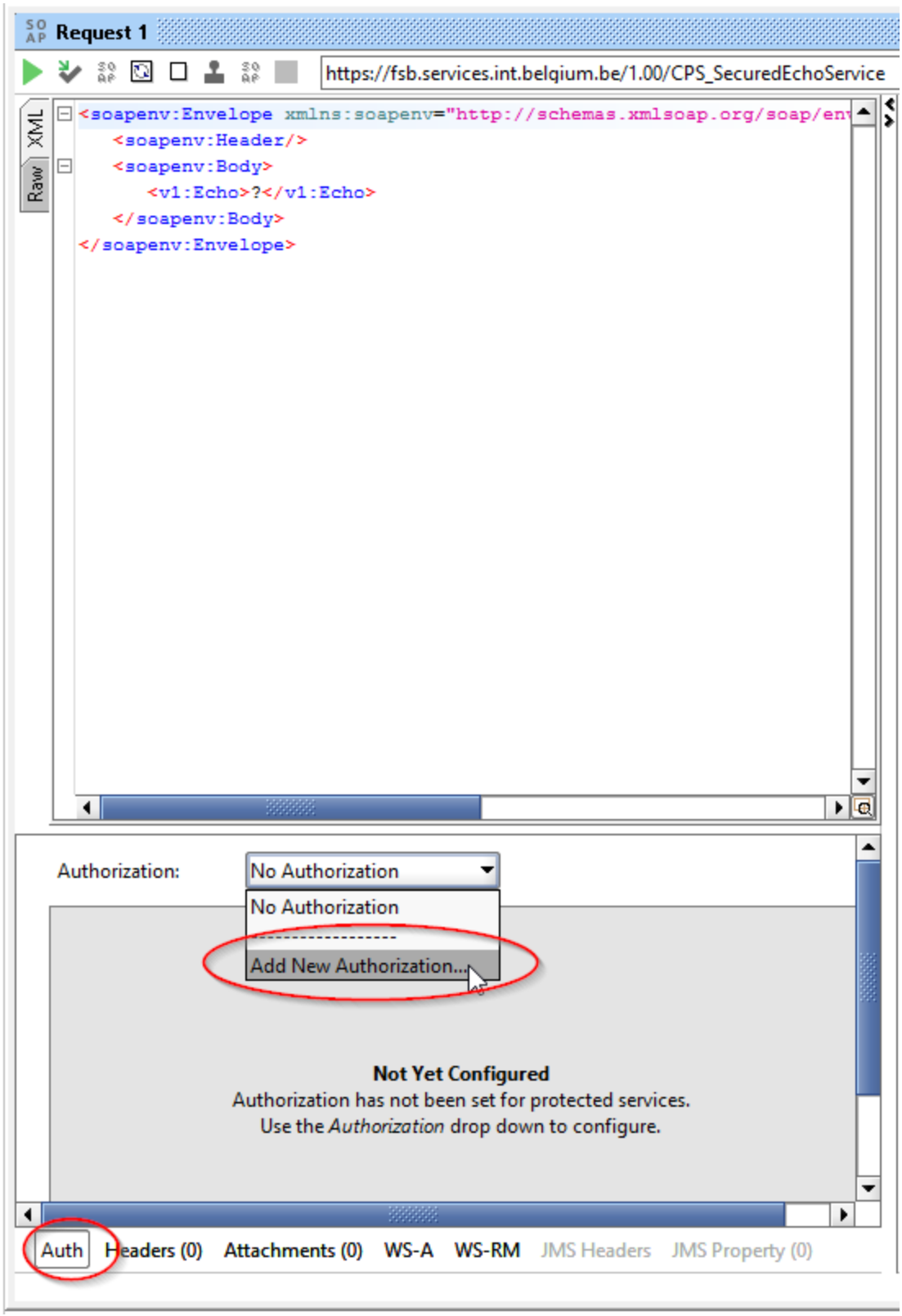
  3= password of private key

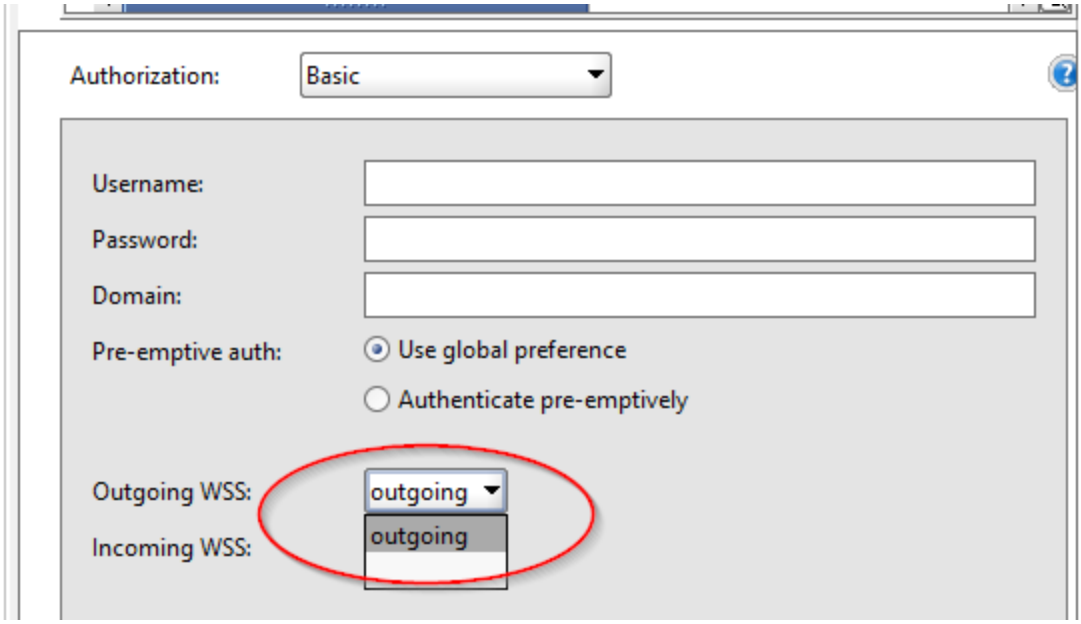
  4= key Identifier type should be set to Binary Security token

  5= mark "Use single certificate for signing"

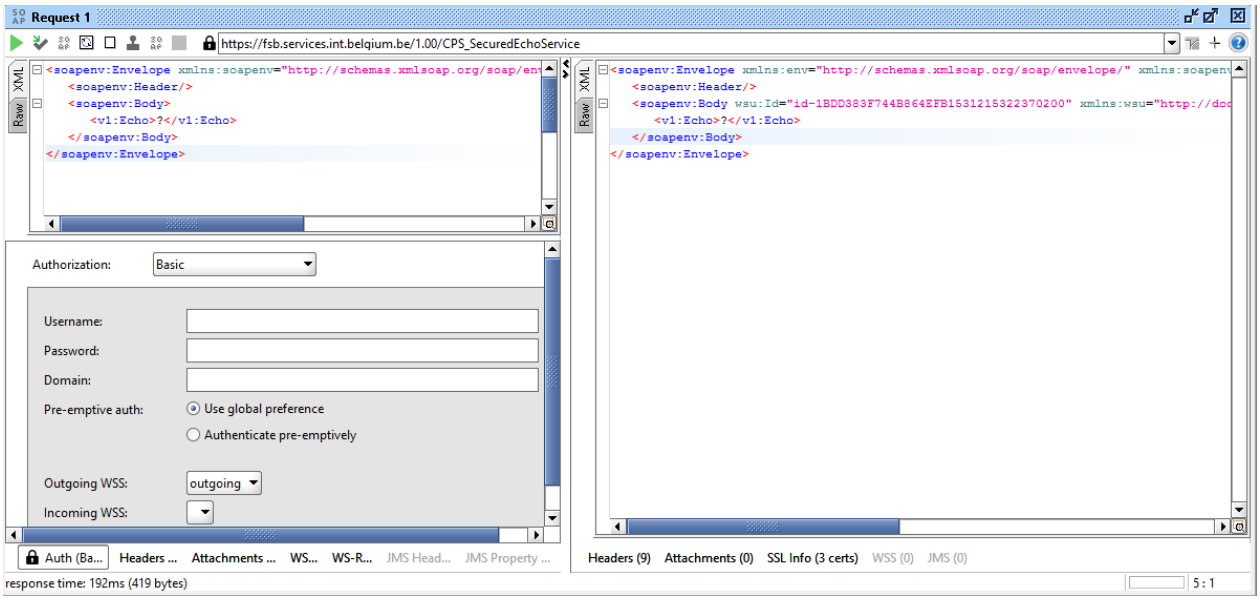


- now, open a request, and apply the security settings. Click on the Auth tab of the request, choose "Add New Authorization", choose Basic, and finally select your outgoing WSS config.


Request 1
https://fsb.services.int.belgium.be/1.00/CPS_SecuredEchoService
XML
Raw
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/en
<soapenv:Header/>
<soapenv:Body>
<v1:Echo>?</v1:Echo>
</soapenv:Body>
</soapenv:Envelope>
Authorization:
No Authorization
No Authorization
Add New Authorization...
Not Yet Configured
Authorization has not been set for protected services.
Use the Authorization drop down to configure.
Auth
Headers (0)
Attachments (0)
WS-A
WS-RM
JMS Headers
JMS Property (0)



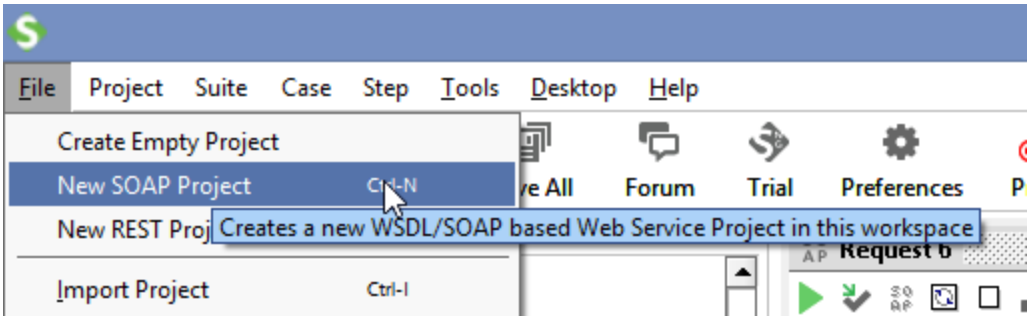
7° Finally click on the green arrow of the request, and you'll get result :



Create SOAP UI project SecuredEchoService 2.0

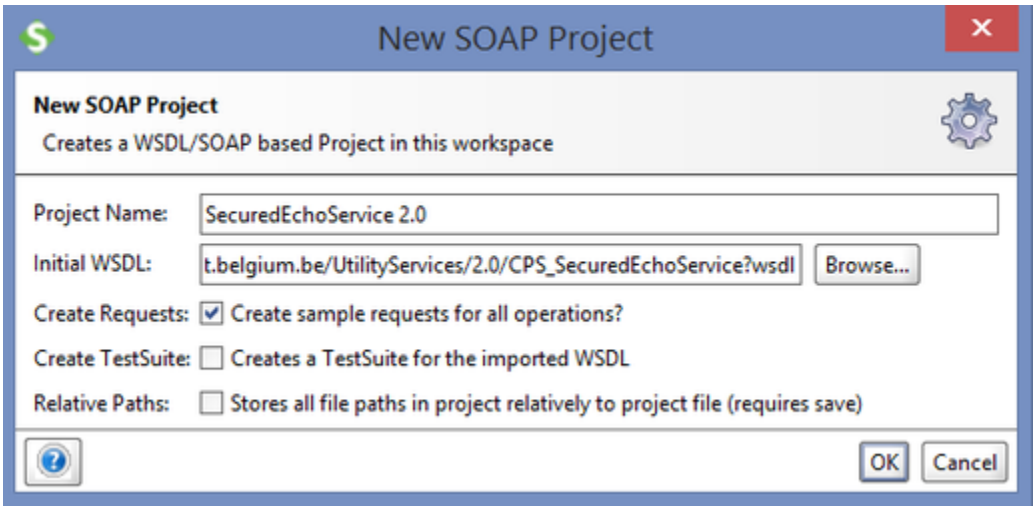
1° Open SOAP UI

2° Select via Menu-File item "New SOAP Project"

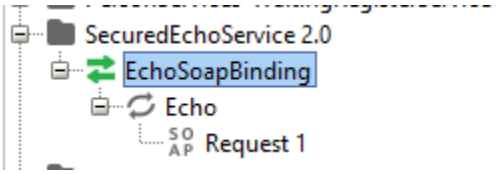


3° Enter "https://fsb.services.int.belgium.be/UtilityServices/2.0/CPS_SecuredEchoService?wsdl" to retrieve the WSDL (or you can download the WSDL from browser, and via button Browse... you can select the WSDL-file). Provide also a Project name.

⚠ When you can't get the WSDL from the URL, there is probably a firewall issue between your platform and FSB.



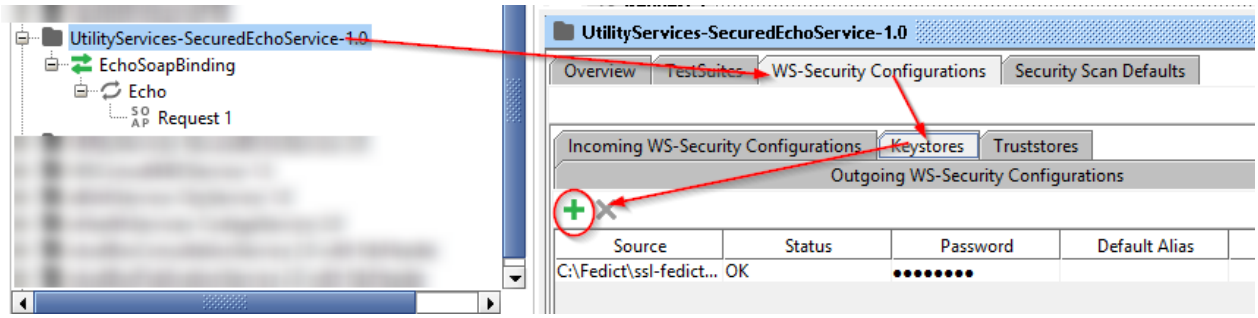
4° You'll see this in the navigator.



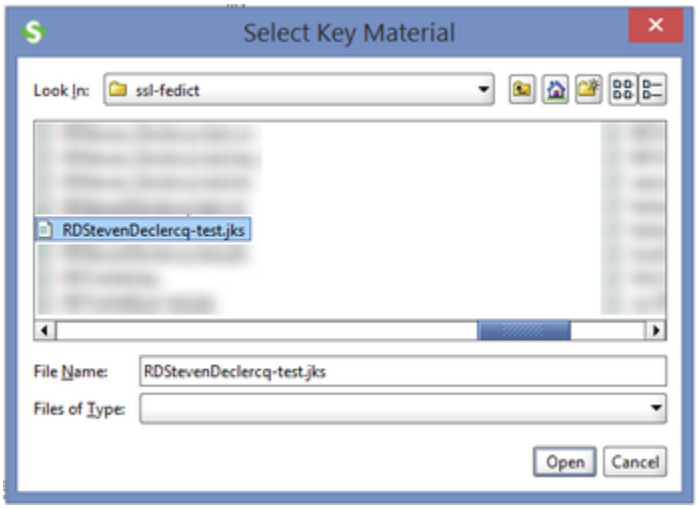
5° As the SecuredEchoService requires a signed request, the next sections explains how to add security

6° Add your own keystore.The keystore contains the private and public key of your certificate, which is required to sign the request.

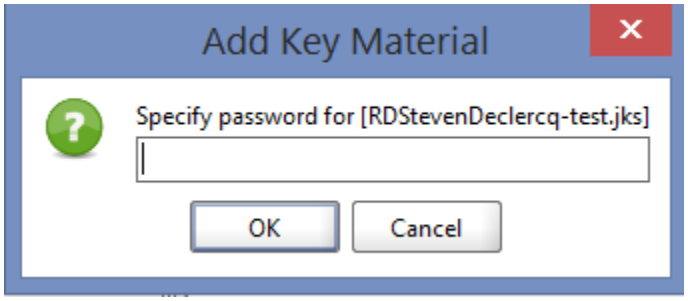
- Double click on the project. This will open the property screen. Go to WS Security Configuration/Keystores
- Click on the green '+' to add your keystore



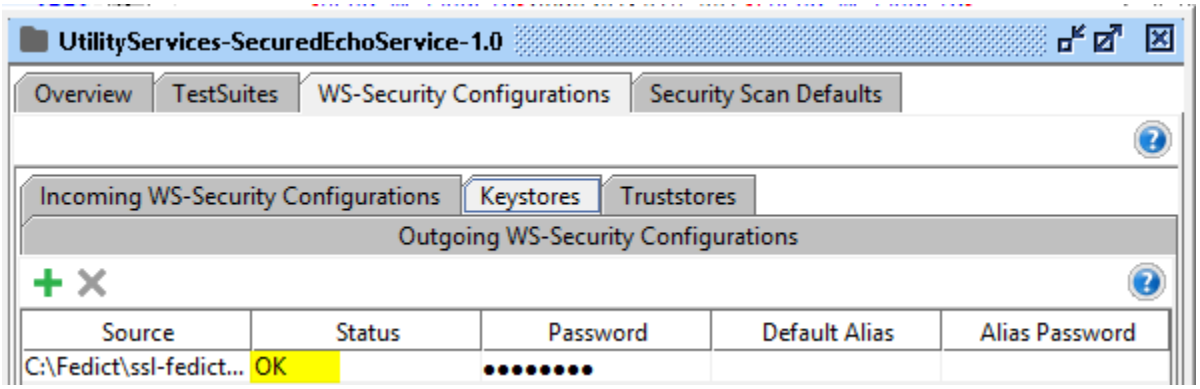
- Most probably, this will be a .jks file that contains the private and public key of your certificate



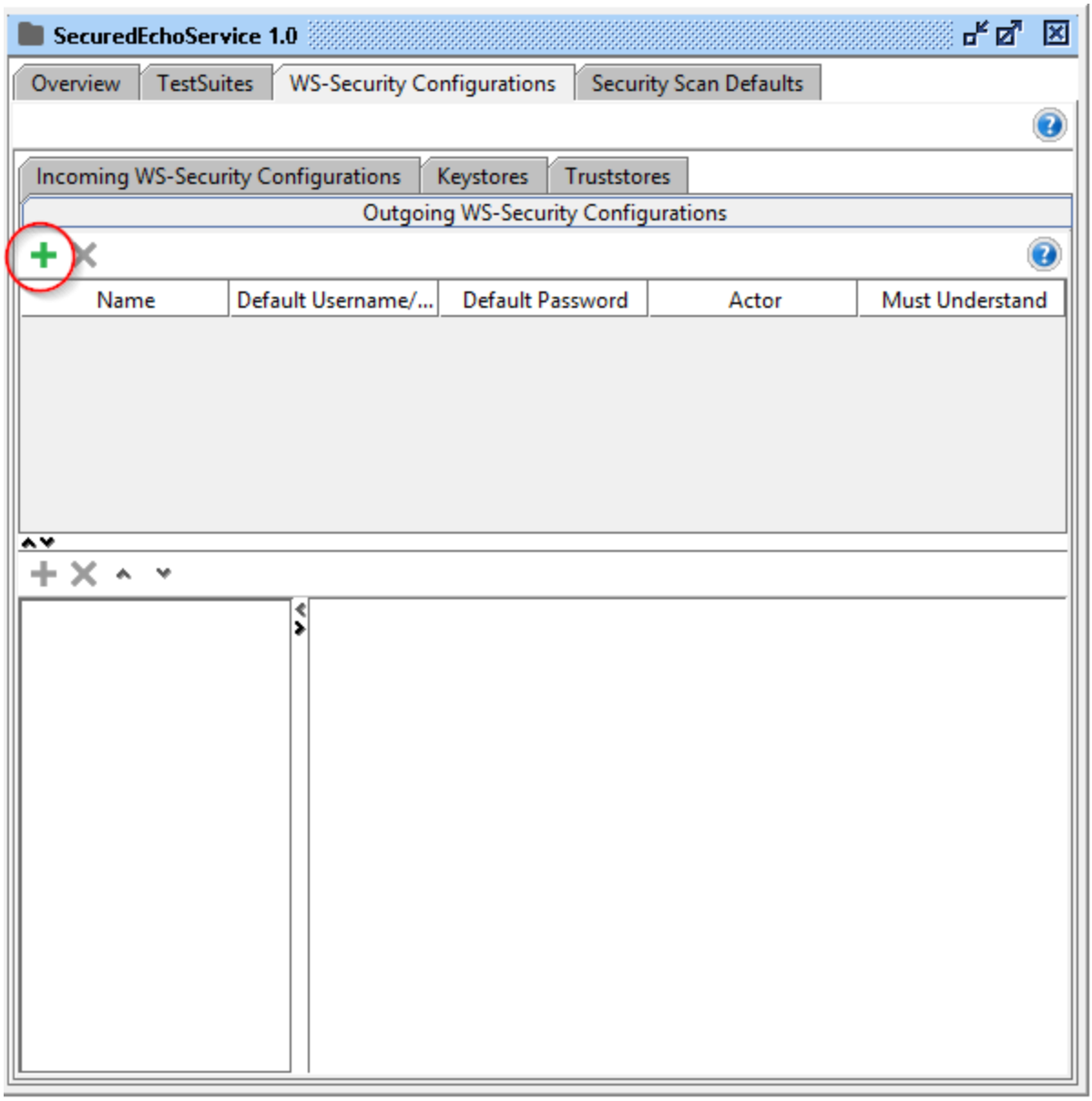
- Enter the password that you have assigned to your keystore



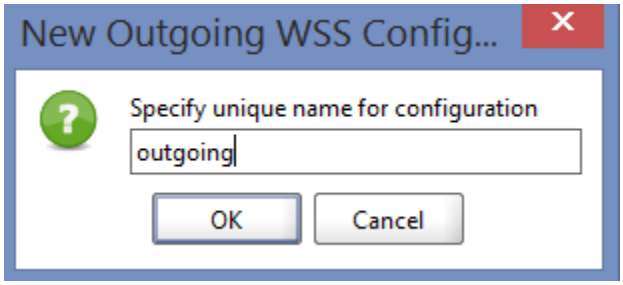
- When the entered password is correct, the status will be 'OK'



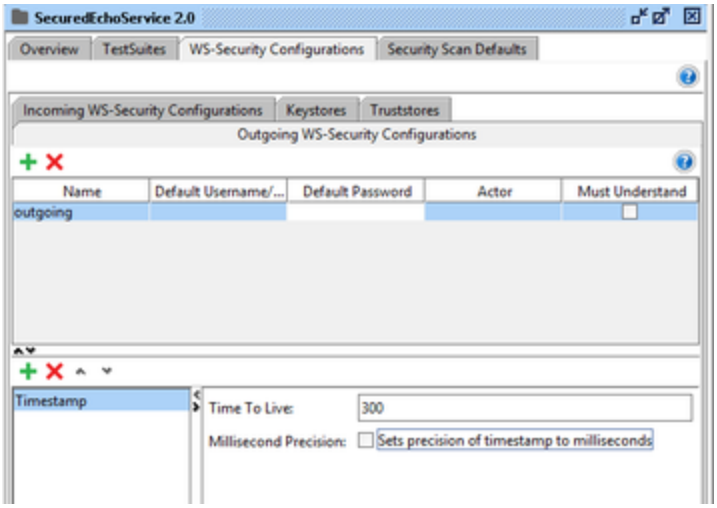
- Now we'll attach your keystore to the siging part of the request. Go to tabpage "Outgoing WS-Security Configurations", click on the green '+ '



- Give the Security Configuration a name (f.e. outgoing)



- Now click on the green '+' to select WSS entries. 2 entries must be applied : Timestamp & Signature . ⚠ The order of the WSS-entries are important. Choose first of all Timestamp and at last Signature
- Set Time To live to 300 and uncheck "Milliseconds Precision"



- Click on the green '+' to select the Signature WSS-entry.
- provide the following information and close this window.

    1= filename of the keystore

    2= alias of private key

    3= password of private key

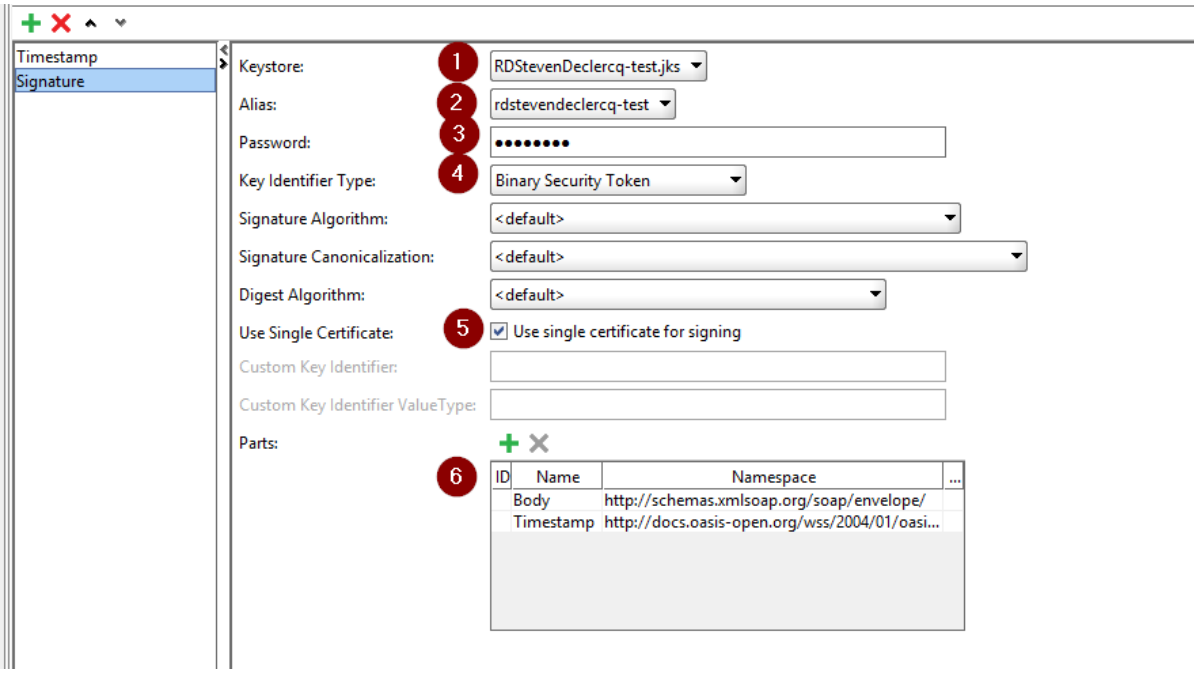
    4= key Identifier type should be set to Binary Security token

    5= mark "Use single certificate for signing"

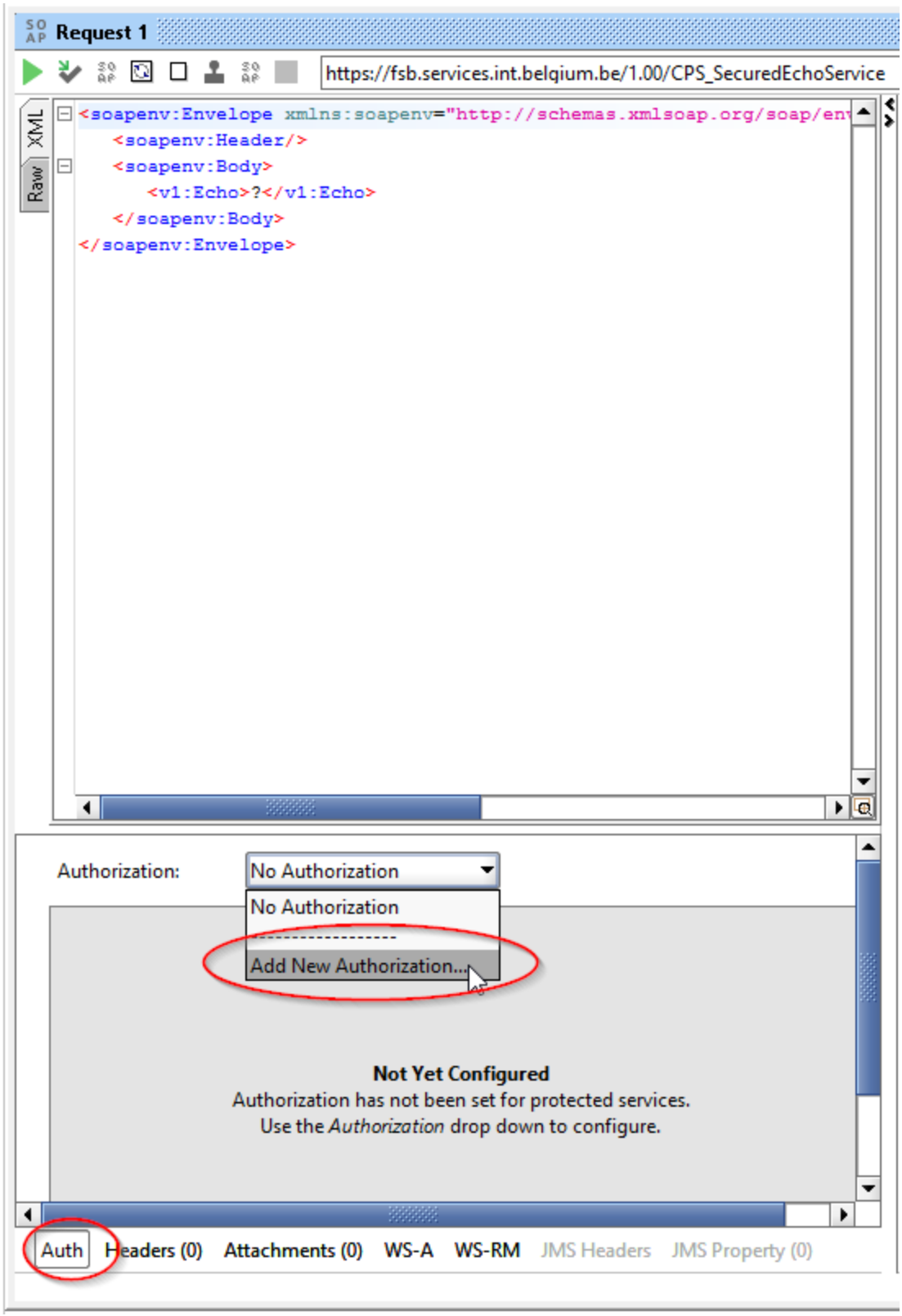
    6= identity which parts should be signed.

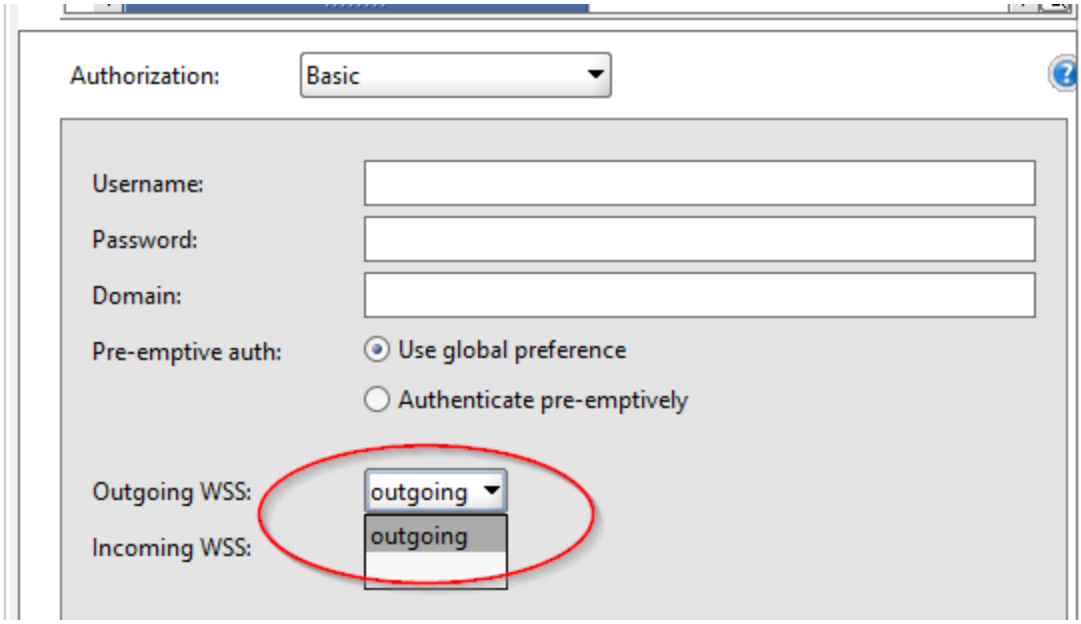
        Body = http://schemas.xmlsoap.org/soap/envelope/ (for SOAP 1.1); http://www.w3.org/2003/05/soap-envelope (for SOAP 1.2)

        Timestamp = http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd

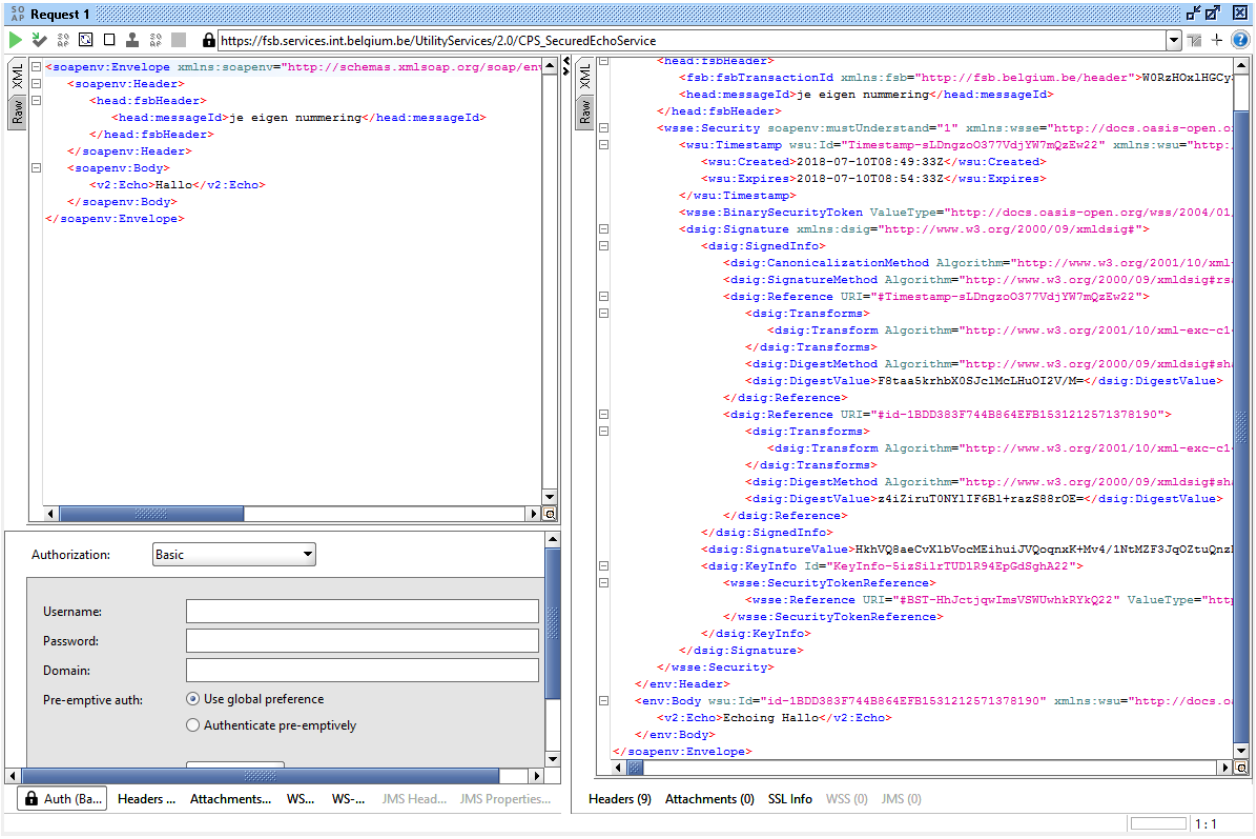


- now, open a request, and apply the security settings. Click on the Auth tab of the request, choose "Add New Authorization", choose Basic, and finally select your outgoing WSS config.


Request 1
https://fsb.services.int.belgium.be/1.00/CPS_SecuredEchoService
XML
Raw
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/en
<soapenv:Header/>
<soapenv:Body>
<v1:Echo>?</v1:Echo>
</soapenv:Body>
</soapenv:Envelope>
Authorization:
No Authorization
No Authorization
Add New Authorization...
Not Yet Configured
Authorization has not been set for protected services.
Use the Authorization drop down to configure.
Auth
Headers (0)
Attachments (0)
WS-A
WS-RM
JMS Headers
JMS Property (0)



7° Finally click on the green arrow of the request, and you'll get result :



Configure downloaded SOAP UI project SecuredEchoService 1.0

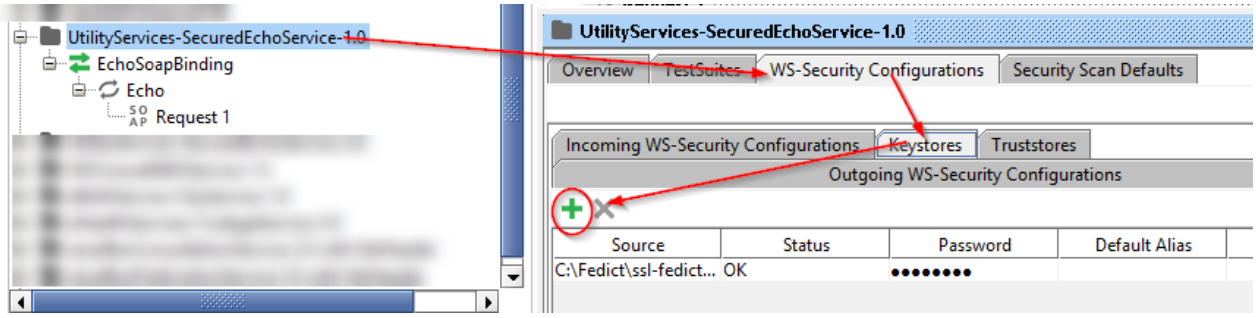
1° Open SOAP UI.

2° Import for example project *SecuredEchoService-soapui-project.xml* from menu 'File/import Project'.

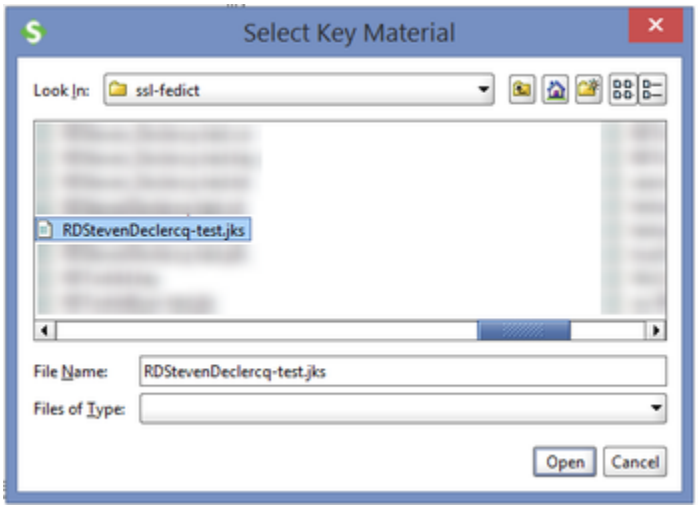
3° During import you might receive error "the project cannot find the keystore". Neglect this message and press continue.

4° Add your own keystore.The keystore contains the private and public key of your certificate, which is required to sign the request.

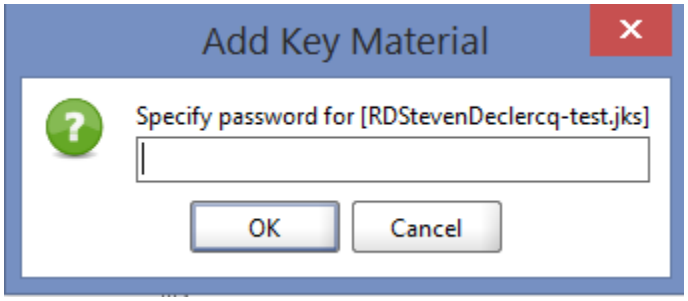
- Double click on the project. This will open the property screen. Go to WS Security Configuration/Keystores
- Click on the green '+' to add your keystore



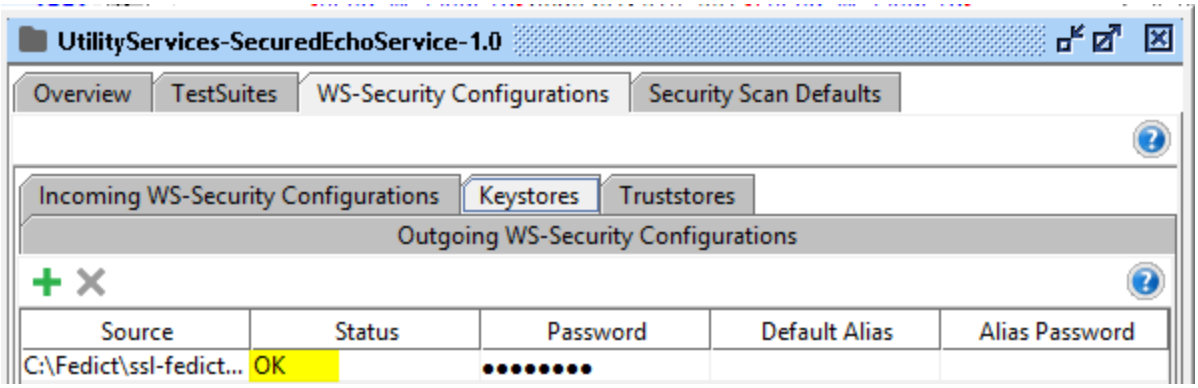
- Most probably, this will be a .jks file that contains the private and public key of your certificate



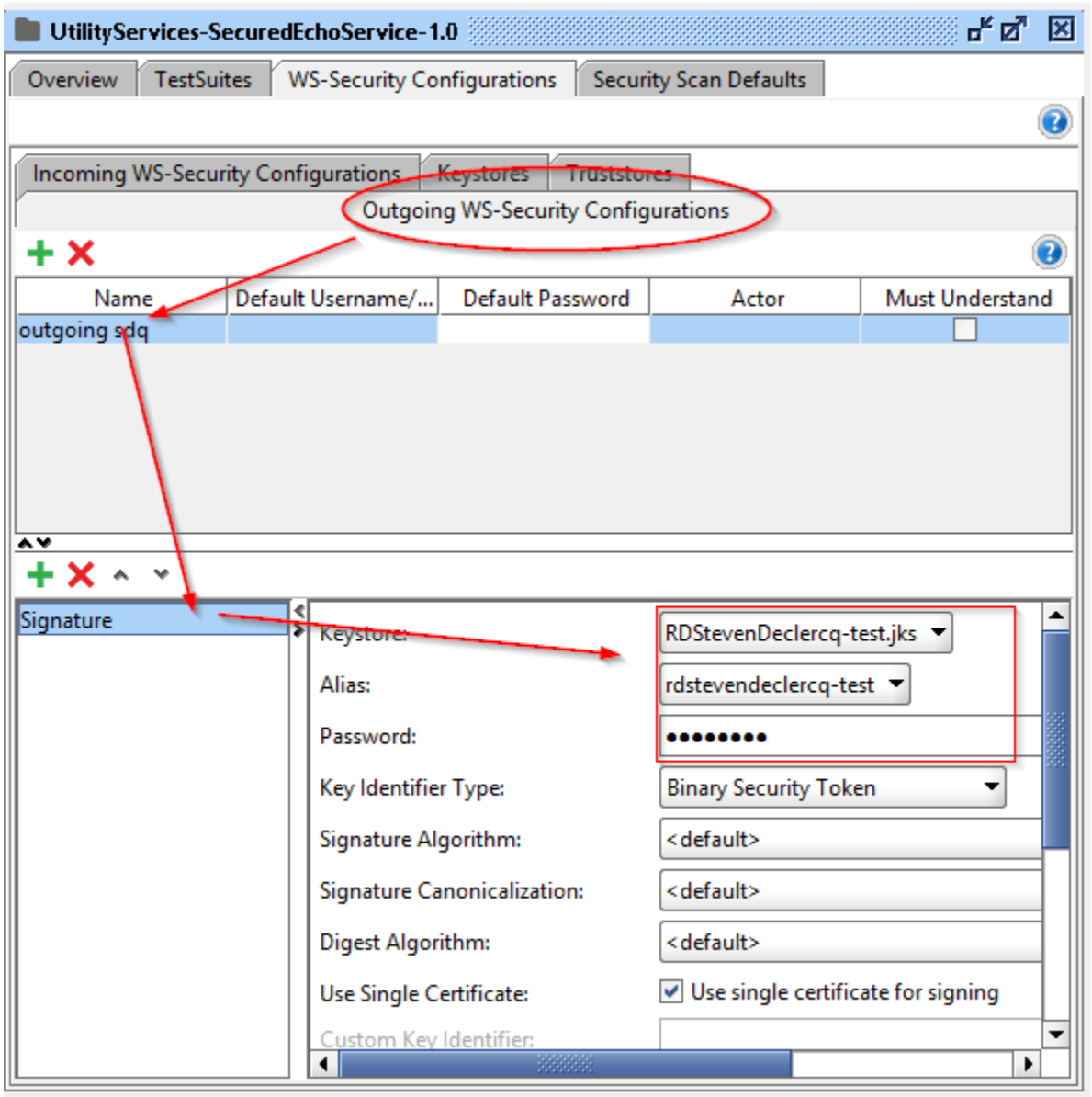
- Enter the password that you have assigned to your keystore



- If the entered password is correct, the status will be 'OK'

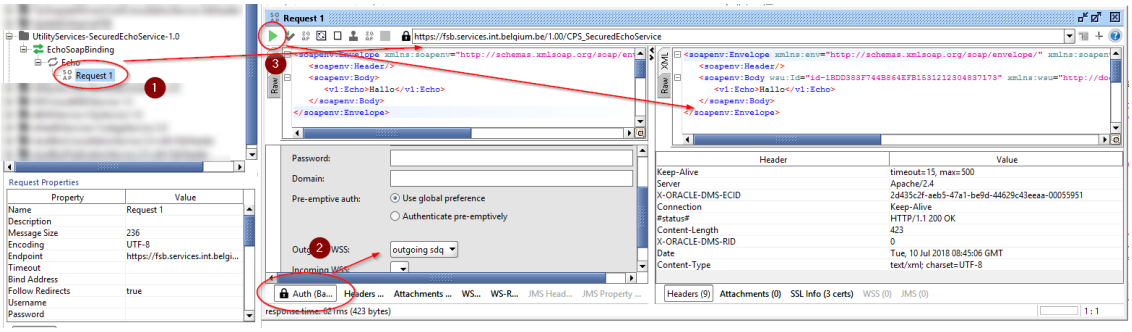


- Now we'll attach your keystore to the siging part of the request.
- Select in the same property popup, tabpage "Outgoing WS-Security Configurations".
- Click on outgoing sdq, then on signature
- Replace the keystore with your own keystore
- Choose the alias of the certificate (every certificate in a keystore has an alias - you'll see all the aliases in the dropdown list)
- Finally provide the password of your private key.



- Leave all the values as they are. ⚠ Only Signature is required for SecuredEchoService 1.0

- Close the popup, and double click on Request 1 (point 1)
  - You'll see an example request containing the value *Hallo* in <Echo>. This value will be echoed back by the SecuredEchoService 2.0
  - Click on Auth (see point 2) to verify that *outgoing sdq* is selected
  - Choose your endpoint (integration https://fsb.services.int.belgium.be/1.0/CPS_SecuredEchoService; production https://fsb.services.pr.belgium.be/1.0/CPS_SecuredEchoService)
  - Finally click on the green flag (execute) (point 3)
    - You will receive a response that contains the echoed value "Hallo"



Configure downloaded SOAP UI project SecuredEchoService 2.0

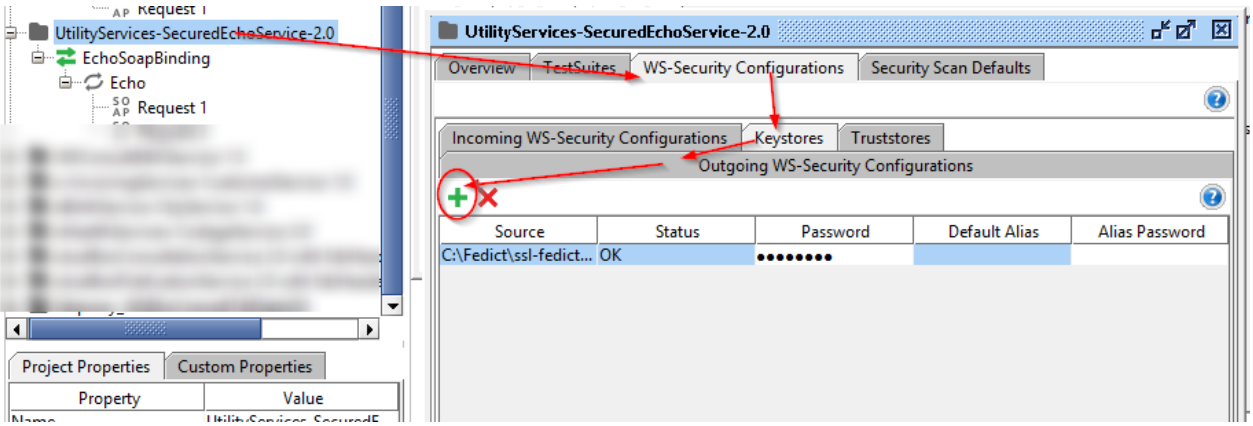
1° Open SOAP UI.

2° Import for example project *SecuredEchoService-with-FsbHeader-soapui-project.xml* from menu 'File/import Project'.

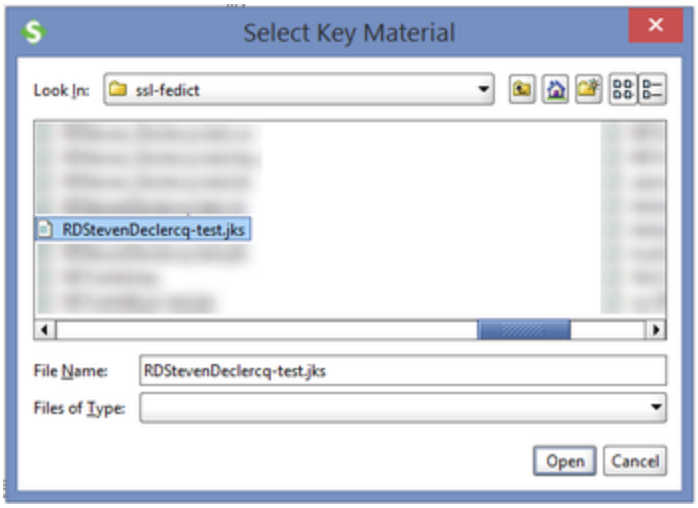
3° During import you might receive error "the project cannot find the keystore". Neglect this message and press continue.

4° Add your own keystore.The keystore contains the private and public key of your certificate, which is required to sign the request.

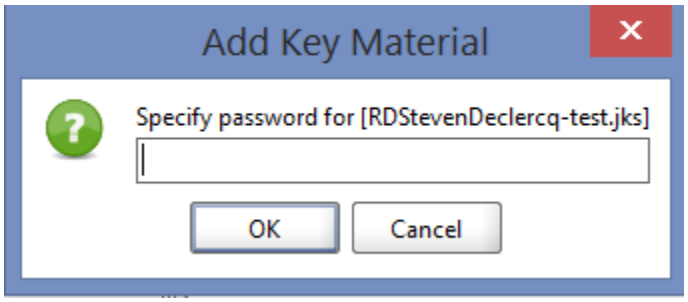
- Double click on the project. This will open the property screen. Go to WS Security Configuration/Keystores
- Click on the green '+' to add your keystore



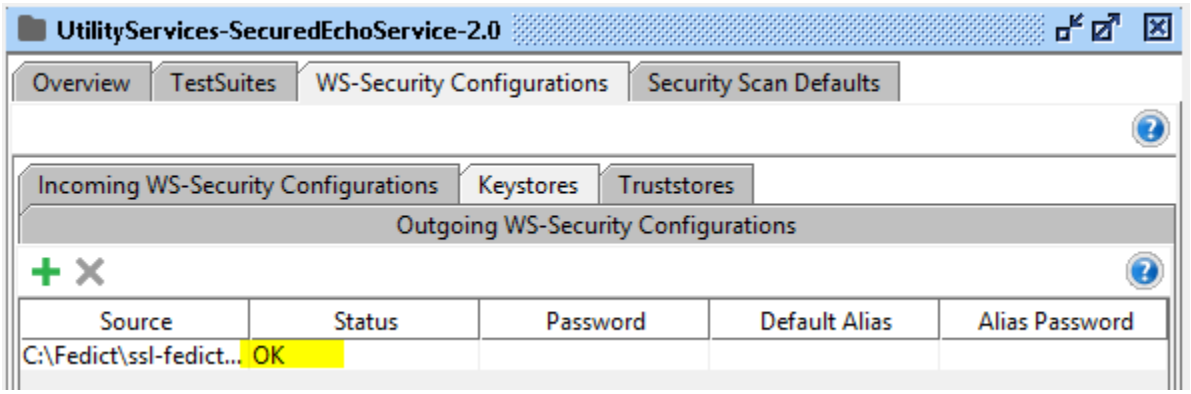
- Most probably, this will be a .jks file that contains the private and public key of your certificate



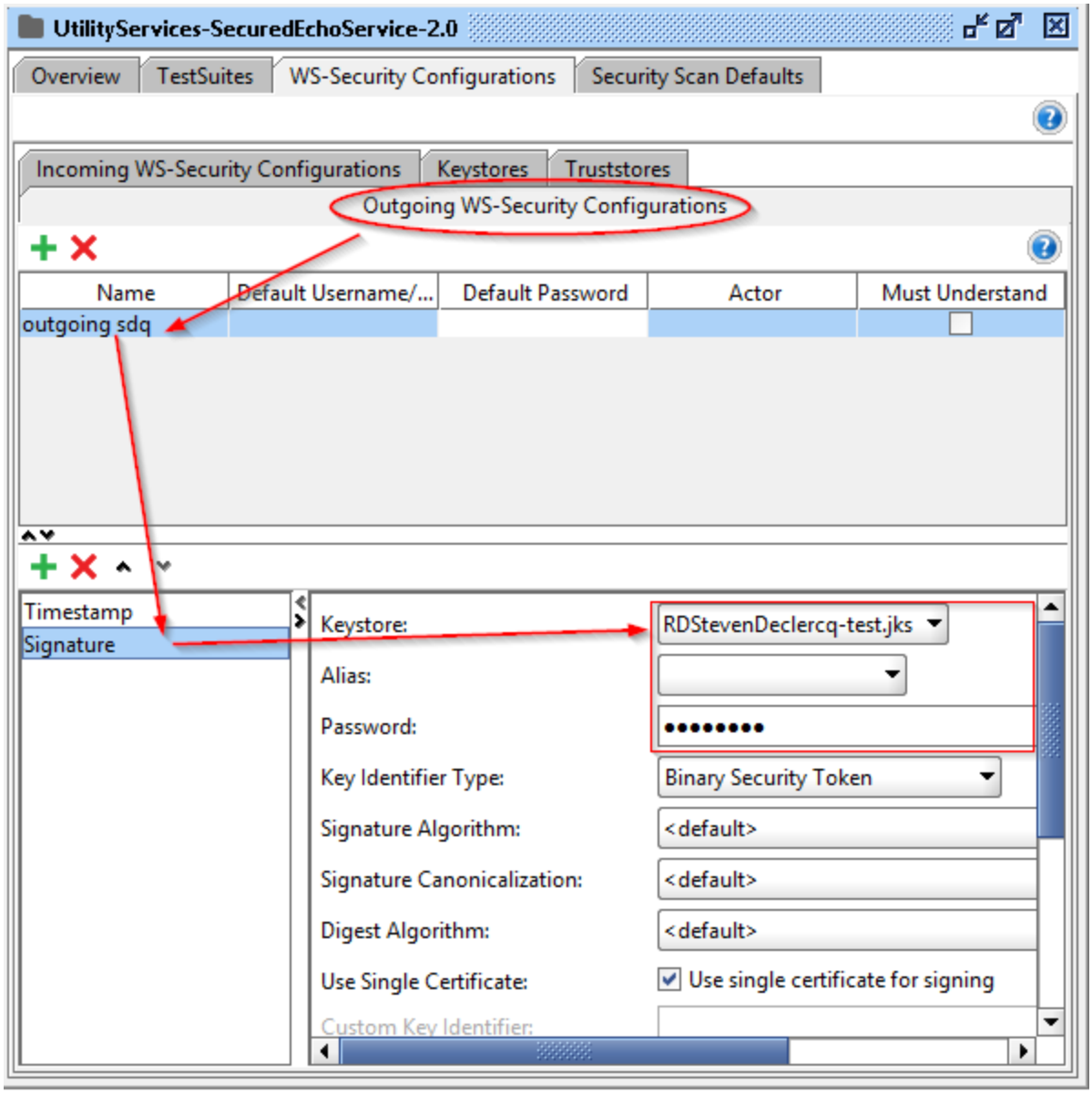
- Enter the password that you have assigned to your keystore



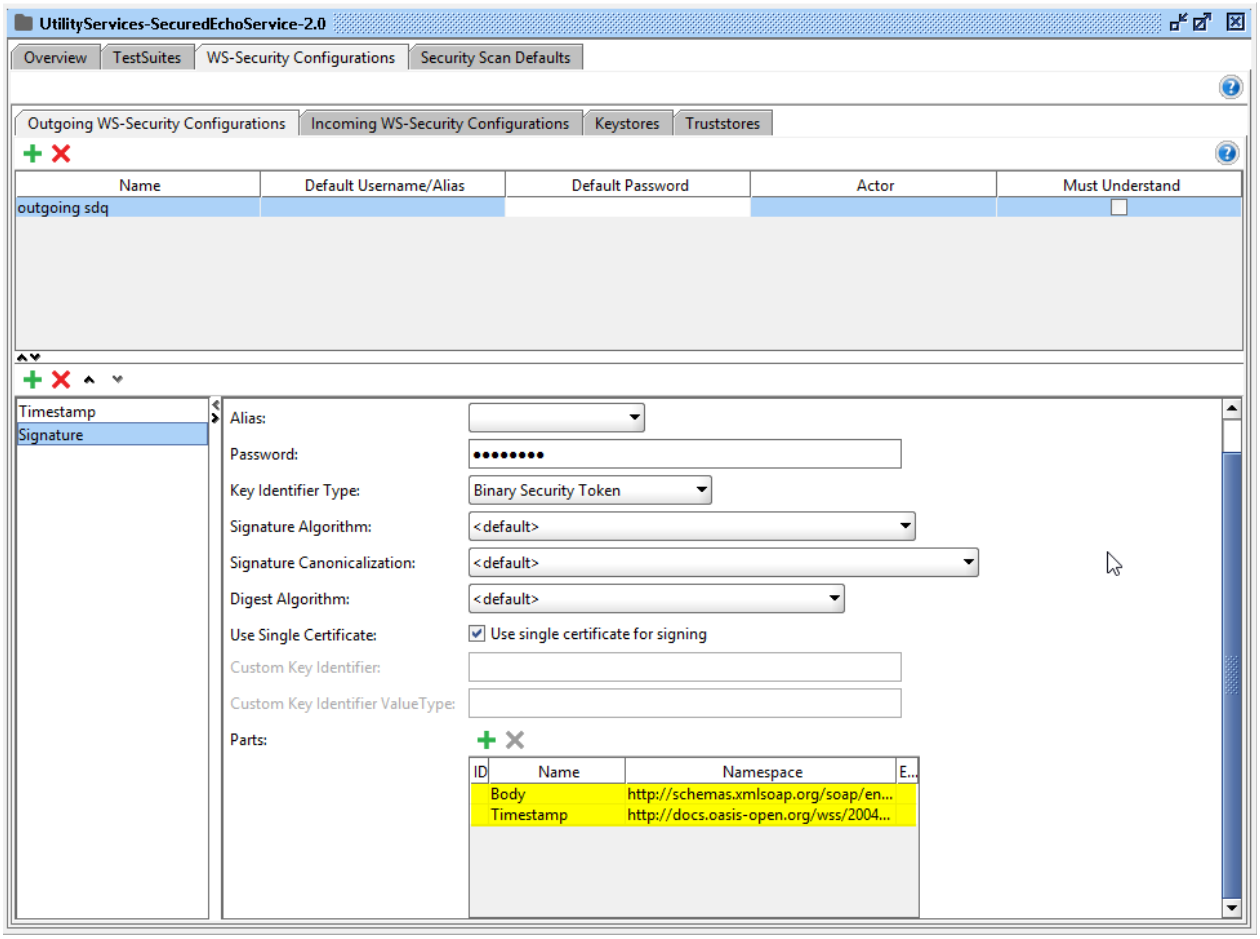
- If the entered password is correct, the status will be 'OK'



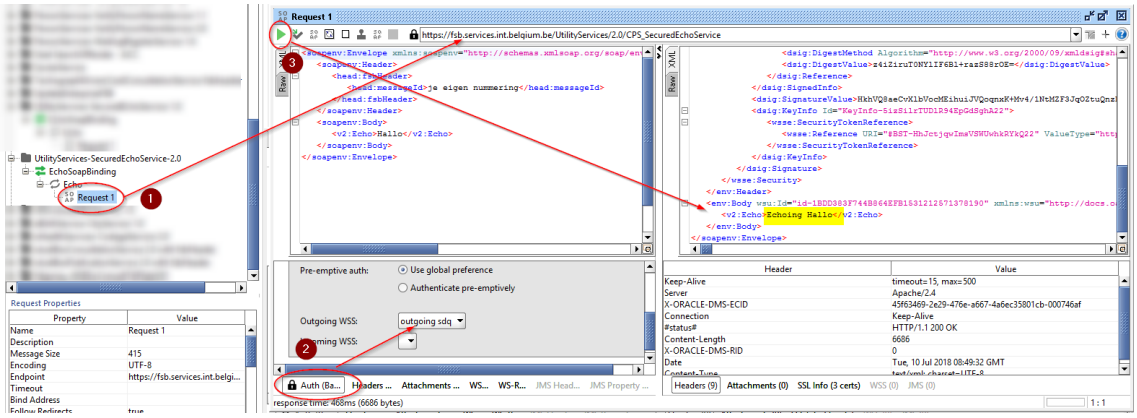
- Now we'll attach your keystore to the siging part of the request.
- Select in the same property popup, tabpage "Outgoing WS-Security Configurations".
- Click on outgoing sdq, then on signature
- Replace the keystore with your own keystore
- Choose the alias of the certificate (every certificate in a keystore has an alias - you'll see all the aliases in the dropdown list)
- Finally provide the password of your private key.



- Leave all the values as they are. Timestamp and Signature are required for SecuredEchoService 2.0; only Signature is required for SecuredEchoService 1.0
- When you scroll down you'll note the signing parts (You'll not see those topics in the SecuredEchoService)



- Close the popup, and double click on Request 1 (point 1)
  - You'll see an example request containing the value *Hallo* in <Echo>. This value will be echoed back by the SecuredEchoService 2.0
  - Click on Auth (see point 2) to verify that *outgoing sdq* is selected
  - Choose your endpoint (integration https://fsb.services.int.belgium.be/UtilityServices/2.0/CPS_SecuredEchoService; production https://fsb.services.pr.belgium.be/UtilityServices/2.0/CPS_SecuredEchoService)
  - Finally click on the green flag (execute) (point 3)
    - You will receive a response (body + timestamp signed) which contains the echoed value "Echoing Hallo"



Problems/Issues

When there are issues with the request, the following response could be returned :

- *OSB-386200: General web service security error*
  - This can have many causes :
    - is the request signed ? check Auth tab-page to verify Outgoing WSS
    - is the password of the private key correct ? (you'll not be notified when this is the case)
    - is the public certificate known and/or uploaded into the LDAP of FSB ?
    - is the timestamp present ?
    - go to the HowTo-part of this document to review all the steps required to have a working SOAP UI Project

- *OSB-386102: Message-level authorization denied*
  - When you receive this message, you need to contact BOSA. This means that the certificate is known @BOSA, but no proper authorisations have been granted to the certificate to use SecuredEchoServices.