Authentication of a Web Service Client via Certificate

Release 650

HELP.BCWEBSERVICES_TUTORIALS

Copyright

Icons in Body Text

Icon	Meaning
⚠	Caution
💬	Example
💡	Note
🧭	Recommendation
syn	Syntax

Additional icons are used in SAP Library documentation to help you identify different types of information at a glance. For more information, see *Help on Help* → *General Information Classes and Information Classes for Business Information Warehouse* on the first page of any version of *SAP Library*.

Typographic Conventions

Type Style	Description
Example text	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options.
	Cross-references to other documentation.
`Example text`	Emphasized words or phrases in body text, graphic titles, and table titles.
EXAMPLE TEXT	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
`Example text`	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
`Example text`	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
`<Example text>`	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
`EXAMPLE TEXT`	Keys on the keyboard, for example, `F2` or `ENTER`.

Authentication of a Web Service Client via Certificate 5
- Importing Projects 6
- Creating and Using Logical Ports 7
- Creating, Signing, and Exporting the Client Certificate 9
- Configuring the Web Service Client for Logon Using a Certificate 11
- Configuring the Web Service Client for Securing the Server Identity (Optional) 14

Authentication of a Web Service Client via Certificate

Task

In this tutorial, you will learn – based on an predetermined project - all the development steps required to configure an existing Web service in such a way that it authenticates itself to the Web service provider via a certificate. The predefined Web service *CreditCheck* is to be used.

For further information about the Web service *CreditCheck*, see the following tutorial: Creating a Web Service.

The Web service *CreditCheck* offers the simple function of checking the credit-worthiness of a customer on the basis of the customer number. If a customer number between 1001 and 1005 is passed, the credit limit of the customer is returned. Otherwise a message informs you that no corresponding customer exists.

The client uses a jsp application to simulate a simple sales portal. If a customer makes an order, the system uses the corresponding customer number to check whether the sum of the order is within the credit limit of that customer. This check is carried out using the Web service *CreditCheck*.

Objectives

By the end of this tutorial, you will be able to:

- ✓ Authenticate an existing Web service client at the Web service provider with a certificate.

Prerequisites

- ☐ You can call the Web service *CreditCheck* successfully by setting up, for example, the tutorial *User Authentication for a Java Web server via Certificate*. Another option would be to deploy the solution EAR file to a J2EE Engine. You can find it under *Samples&Tutorials* in the corresponding directory (/ears/strong).
- ☐ All systems involved are configured for the use of SSL. You can find information concerning this topic here: Deploying the SAP Java Cryptographic Toolkit

Systems, Installations, and Authorizations

- ☐ You have installed the SAP NetWeaver Developer Studio and configured it for use of the SAP J2EE Engine.
- ☐ You can access the SAP J2EE Engine using the Visual Administrator.

Knowledge

- ☐ You understand the operation mode of a JSP Web service client, for example, by studying the topic Creating a Client JavaServer Page.

Next Step:

Importing Projects [Page 6]

Importing Projects

The following projects are available for this tutorial:

- Deployable proxy project *CreditLimitCheckProxy (the proxy used for the Web service CreditLimitCheck)*
- WEB project *CreditLimitCheckWeb* (the Web service consumer)
- EAR project *CreditLimitCheckWebEAR* (for deploying the Web project)

The projects are available in the file JAVA_CLIENT_JAVA_RAW.zip. To download the file, click here.

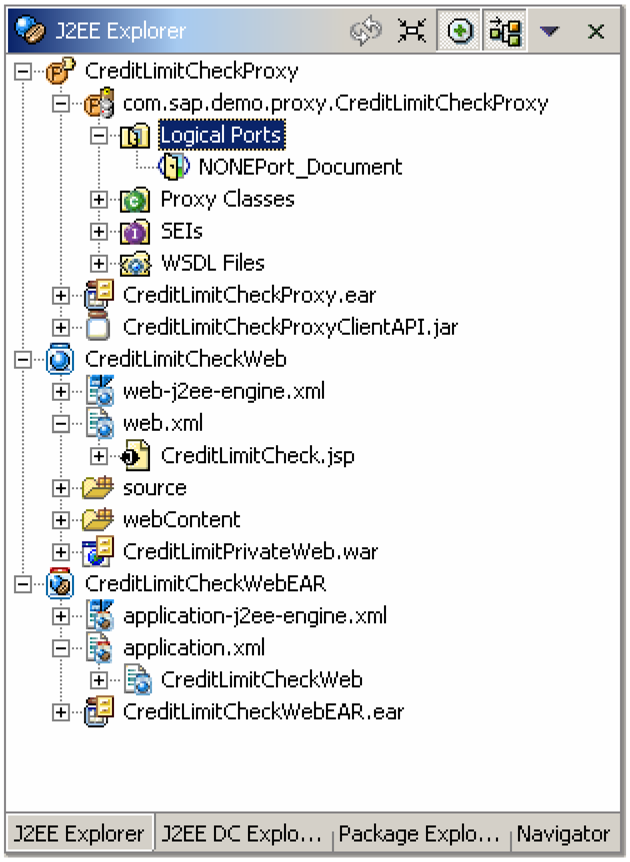
Procedure

Importing Projects into the SAP NetWeaver Developer Studio

1. Download the ZIP file JAVA_CLIENT_JAVA_RAW.ZIP, which contains the projects *CreditLimitCheckProxy*, *CreditLimitCheckWeb*, and *CreditLimitCheckWebEAR,* and save the ZIP file to any directory on your local hard disk or directly in the work area of the SAP NetWeaver Developer Studio.
2. Unpack the ZIP file.
3. Start the SAP NetWeaver Developer Studio.
4. Import the projects *CreditLimitCheckProxy, CreditLimitCheckWeb* and *CreditLimitCheckEAR* and open the J2EE Explorer in the J2EE deployment perspective.

Result

Once you have imported the project into the Developer Studio, the following structure is displayed in the *J2EE Explorer*.



Next Step:

Creating and Using Logical Ports [Page 7]

Creating and Using Logical Ports

A logical port contains the configuration of the client-side SOAP runtime environment, such as the URL or the security settings of a Web service. If you want to access a Web service using a proxy, this is done via a logical port. Every logical port corresponds to a configuration of the Web service as offered by the Web service provider.

To be able to create a logical port, you first require a Web service definition. The latter can be obtained using the overview page of a Web service, for example. However, it can also be imported from the local file system or a UDDI registry.

This section shows you how, starting from an existing proxy, you can create a logical port and access it from within the application.

Prerequisites

☐ You have ensured that your Workbench proxy settings are the same as those in your Intranet. Check these settings in the Developer Studio by choosing *Window* → *Preferences* → *Workbench*

→ *Proxy Settings*.

☐ The structure of your projects *CreditLimitCheckProxy*, *CreditLimitCheckWeb,* and *CreditLimitCheckWebEar* is currently displayed in the *J2EE Explorer*.

Procedure

Creating and Using Logical Ports

1. Display the overview page on the home page of the Web service *CreditLimitPrivateLocal.* The link to the Web service definition can be found under the heading *WSDL*. Copy this link to your clipboard and return to the SAP NetWeaver Developer Studio.
2. To create the logical ports, choose *New – Logical Ports* from the context menu for the node *CreditLimitCheckProxy – com.sap.demo.proxy.CreditCheckLocalProxy – Logical Ports*. The window *WSDL URL* appears.
3. In the field *Wsdl*, insert the link from the clipboard and confirm by choosing *OK*. The logical port *STRONGPort_Document* appears in your project hierarchy.

   If the specified logical port should not appear, check whether you are using the Web service definition of the correct Web service. Also, note the prerequisites for this tutorial.
4. Open the file CreditLimitCheck.jsp by doubleclicking the node *CreditLimitCheckWeb – web.xml – CreditLimitCheck.jsp.* Then activate the tab page *Source* (at bottom left in your preview pane) to display the source text.
5. Search for the following statement in the source code *obj.getLogicalPort("NONEPort_Document", CreditCheckViDocument.**class**);* replace it by *obj.getLogicalPort("STRONGPort_Document", CreditCheckViDocument.**class**);*. Save your changes.

```
CreditCheckViDocument port = (CreditCheckViDocument)
    obj.getLogicalPort("STRONGPort_Document",
    CreditCheckViDocument.class);
```

   The Web service consumer now uses the logical port *STRONGPort_Document,* which accesses a configuration of the Web service that requires authentication via a certificate.

Deploying the Changed Web Service and Proxy

1. Choose *Build EAR* from the context menu for the project *CreditLimitCheckProxy* and deploy the file *CreditLimitCheckProxy – CreditLimitCheckProxy.ear* on the J2EE Engine.
2. Regenerate the files *CreditLimitCheckWeb.war* and *CreditLimitCheckWebEar.ear* by choosing *Build WEB Archive* or *Build Application Archive* from the context menus for the projects *CreditLimitCheckWeb* and *CreditLimitCheckWebEar* and deploy the file *CreditLimitCheckWebEAR.ear* on your J2EE Engine.

Result

Based on the consumer of the Web service CreditLimitCheck you have created a new logical port called *STRONGPort_Document* and have made this accessible by deploying the proxy on the Engine. The Web service is now configured in such a way that it uses the port *STRONGPort_Document*, which accesses a configuration of the Web service that requires user authentication via a certificate.

Next Step:

Creating, Signing and Exporting the Client Certificate [Page 9]

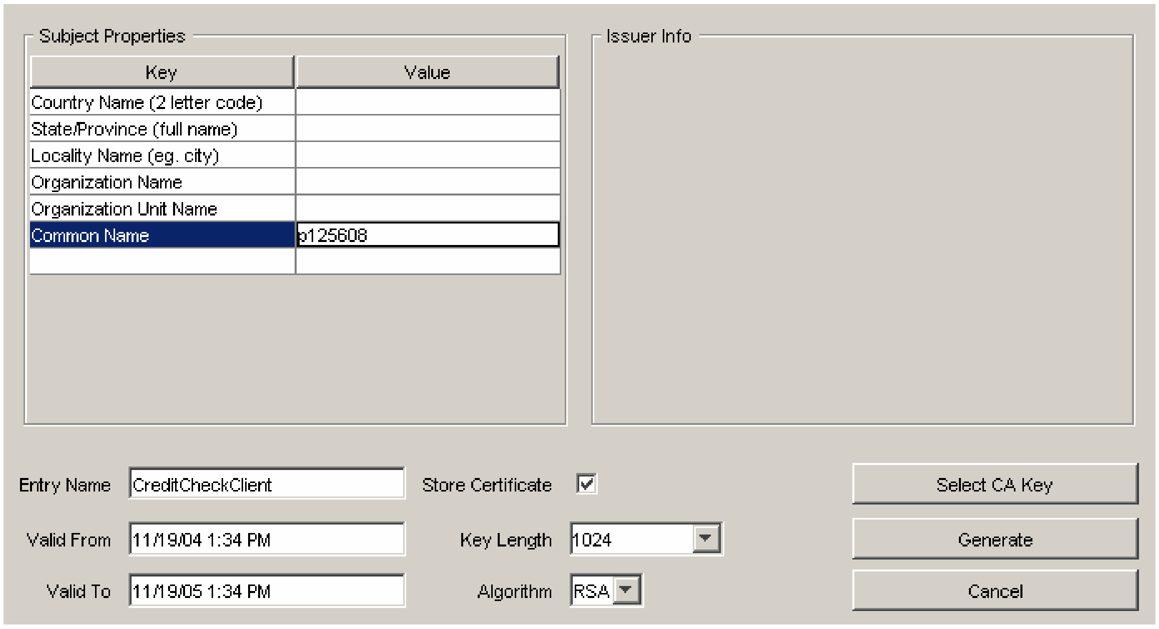
Creating, Signing, and Exporting the Client Certificate

Since the Web service provider requires a user authentication using a certificate, we must create this certificate. The certificate must be issued on the Web service client and be signed by a CA considered trustworthy by the Web service provider.

Procedure

Creating and Signing the Certificate

1. Start the Visual Administrator and log on to the J2EE Engine on which you deployed the Web service client application.
2. Select the node <*SID*> - *Server … - Services – Key Storage* (where <SID> is the system ID of the client J2EE) and activate the tabs *Runtime* (top) and *Data* (bottom) in the right window.
3. Create a new keystore view by choosing the *Create View* button below the *Views* area. Give the new view a descriptive name - for example, *WebServiceClient* - and confirm your entry. The new entry is displayed in the *Views* area.
4. Select your newly created keystore view.
5. In the *Entry* section (bottom right), choose *Create*. The window *Key and Certificate Generation* appears.
6. Make the appropriate entries in the fields under *Subject Properties*. The only crucial value is that of the *Common Name* key. It must correspond to the fully-qualified name of the used client domain.
7. In the *Entry Name* field, enter a descriptive name for the certificate (for example, CreditCheckClient) Check the *Store Certificate* option and leave all the other settings unchanged.



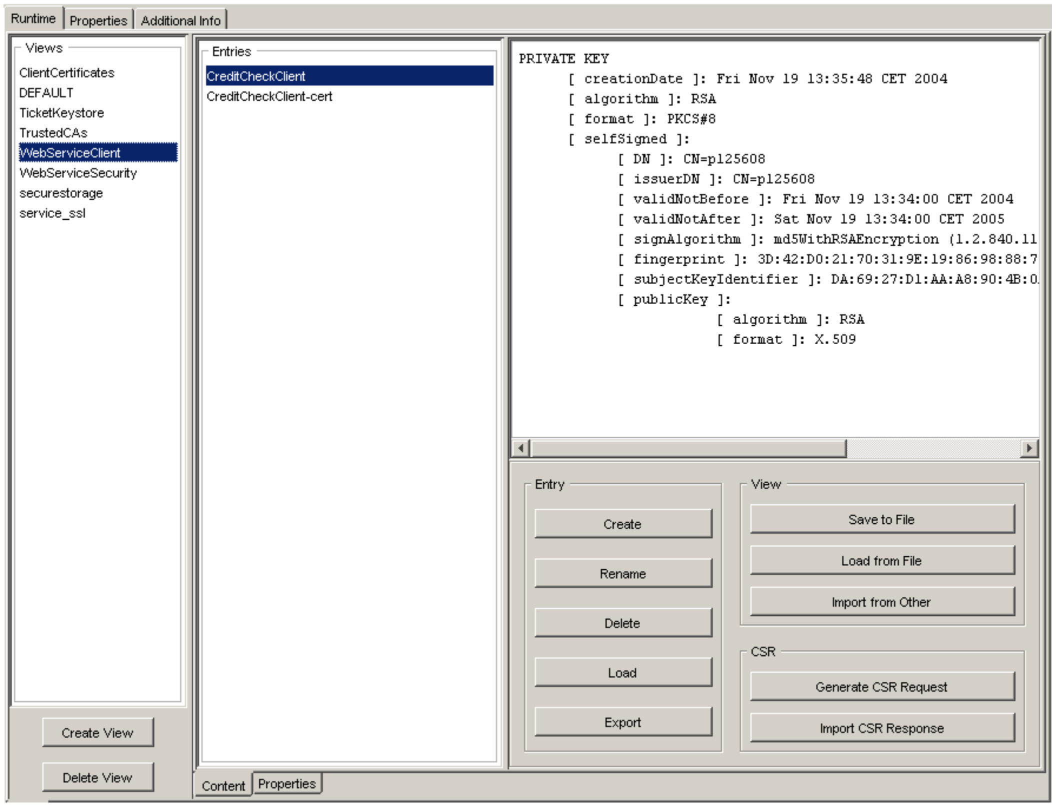
Subject Properties

Key	Value
Country Name (2 letter code)	
State/Province (full name)	
Locality Name (eg. city)	
Organization Name	
Organization Unit Name	
Common Name	p125608

Issuer Info

Entry Name: CreditCheckClient
Valid From: 11/19/04 1:34 PM
Valid To: 11/19/05 1:34 PM
Store Certificate: ✓
Key Length: 1024
Algorithm: RSA

8. Choose *Generate*. Two new entries are now displayed in the *Entries* section.



Entries: CreditCheckClient, CreditCheckClient-cert

```
PRIVATE KEY
      [ creationDate ]: Fri Nov 19 13:35:48 CET 2004
      [ algorithm ]: RSA
      [ format ]: PKCS#8
      [ selfSigned ]:
            [ DN ]: CN=p125608
            [ issuerDN ]: CN=p125608
            [ validNotBefore ]: Fri Nov 19 13:34:00 CET 2004
            [ validNotAfter ]: Sat Nov 19 13:34:00 CET 2005
            [ signAlgorithm ]: md5WithRSAEncryption (1.2.840.11
            [ fingerprint ]: 3D:42:D0:21:70:31:9E:19:86:98:88:7
            [ subjectKeyIdentifier ]: DA:69:27:D1:AA:A8:90:4B:0
            [ publicKey ]:
                        [ algorithm ]: RSA
                        [ format ]: X.509
```

Creating a Request to the CA

1. In the Visual Administrator, select the node *<SID> - Server … - Services – Key Storage* and activate the tabs *Runtime* (top) and *Data* (bottom) in the right window.
2. Select the entry *WebServiceClient* in the *Views* area and the entry *CreditCheckClient* under *Entries*.

3. Choose *Generate CSR Request* in the *CSR area* (bottom right) and, in the dialog box that appears, save the request as a csr file in your file system. Send this file to your CA for signing.

Importing the CA Response

1. In the Visual Administrator, select the node *<SID> - Server … - Services – Key Storage* and activate the tabs *Runtime* (top) and *Data* (bottom) in the right window.
2. Select the entry *WebServiceClient* in the *Views* area and the entry *CreditCheckClient* under *Entries*.

In the *CSR* area (bottom right), choose *Import CSR Response* and select the file you received from the CA as a response to your request.

Exporting the Certificate

1. In the Visual Administrator, select the node *<SID> - Server … - Services – Key Storage* and activate the tabs *Runtime* (top) and *Data* (bottom) in the right window.
2. Select the entry *WebServiceClient* in the *Views* area and the entry *CreditCheckClient-cert* under *Entries*.
3. Choose *Export* in the area *Entry* (bottom right) and, in the dialog box that appears, save the certificate as a crt file in your file system.

Result

You have created a valid certificate and had it signed by a CA. It was stored in the file system as a crt file and can be used to authenticate your SAP J2EE Engine to other systems.

Next Step:

Configuring the Web Service Client for Logon Using a Certificate [Page 11]

Configuring the Web Service Client for Logon Using a Certificate

The certificate with which the client is to log on to the Web service provider was stored in a new keystore view. However, to allow the client application access to it, the authorizations for this object must be adjusted accordingly. Furthermore, the client must be configured in such a way that it uses the intended certificate for logon.

Prerequisites

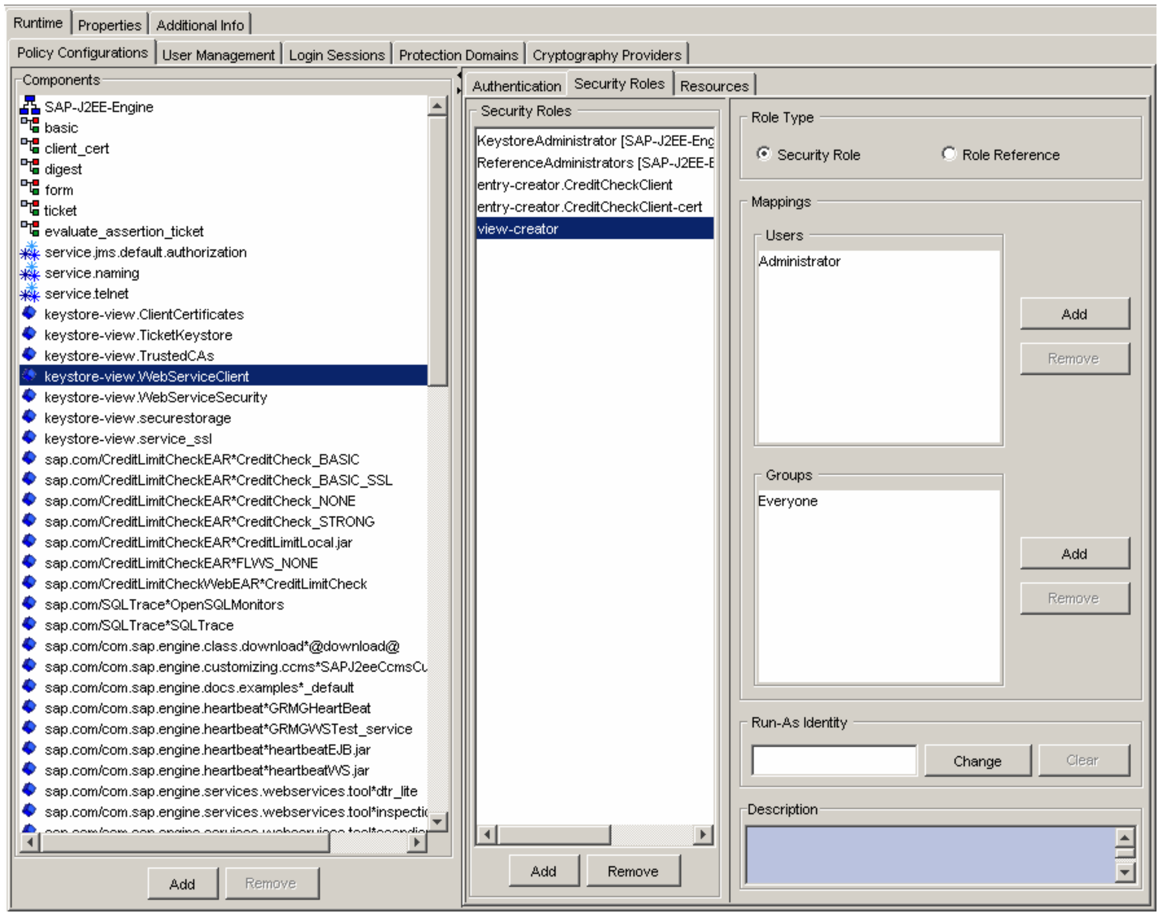
☐ Both the Web service proxy and the Web service client are deployed on a SAP J2EE.

Procedure

Adjusting Authorizations for Keystore View

1. Start the Visual Administrator and log on to the SAP J2EE Engine of the client.

2. Select the node <*SID*> - *Server … - Services – Security Provider*, where <SID> is the system ID of your client J2EE.
3. In the right window, activate the tabs *Runtime* and *Policy Configurations.*
4. In the *Components* area, select the keystore view that contains the certificate to be used (for example, **keystore-view.WebServiceClient**).
5. Activate the tab *Security Roles* (on the right). The authorization configurations for the selected keystore view are displayed.
6. Select the entry **view-creator** from the list under *Security Roles*.
7. Choose *Add* next to the *Groups* area (bottom right). The window *Choose Users or Groups* appears.
8. Activate the *Tree* tab (at the bottom). Select the entry *Everyone* from the *User Tree* and confirm by choosing *OK*. The entry is now displayed under *Groups* and every user of your system is authorized to use the certificates of the corresponding keystore view.



Components:
SAP-J2EE-Engine
basic
client_cert
digest
form
ticket
evaluate_assertion_ticket
service.jms.default.authorization
service.naming
service.telnet
keystore-view.ClientCertificates
keystore-view.TicketKeystore
keystore-view.TrustedCAs
keystore-view.WebServiceClient
keystore-view.WebServiceSecurity
keystore-view.securestorage
keystore-view.service_ssl
sap.com/CreditLimitCheckEAR*CreditCheck_BASIC
sap.com/CreditLimitCheckEAR*CreditCheck_BASIC_SSL
sap.com/CreditLimitCheckEAR*CreditCheck_NONE
sap.com/CreditLimitCheckEAR*CreditCheck_STRONG
sap.com/CreditLimitCheckEAR*CreditLimitLocal.jar
sap.com/CreditLimitCheckEAR*FLWS_NONE
sap.com/CreditLimitCheckWebEAR*CreditLimitCheck
sap.com/SQLTrace*OpenSQLMonitors
sap.com/SQLTrace*SQLTrace
sap.com/com.sap.engine.class.download*@download@
sap.com/com.sap.engine.customizing.ccms*SAPJ2eeCcmsCu
sap.com/com.sap.engine.docs.examples*_default
sap.com/com.sap.engine.heartbeat*GRMGHeartBeat
sap.com/com.sap.engine.heartbeat*GRMGWSTest_service
sap.com/com.sap.engine.heartbeat*heartbeatEJB.jar
sap.com/com.sap.engine.heartbeat*heartbeatWS.jar
sap.com/com.sap.engine.services.webservices.tool*dtr_lite
sap.com/com.sap.engine.services.webservices.tool*inspectio

Security Roles:
KeystoreAdministrator [SAP-J2EE-Eng
ReferenceAdministrators [SAP-J2EE-E
entry-creator.CreditCheckClient
entry-creator.CreditCheckClient-cert
view-creator

Role Type: [✓ Security Role] Role Reference

Users: Administrator

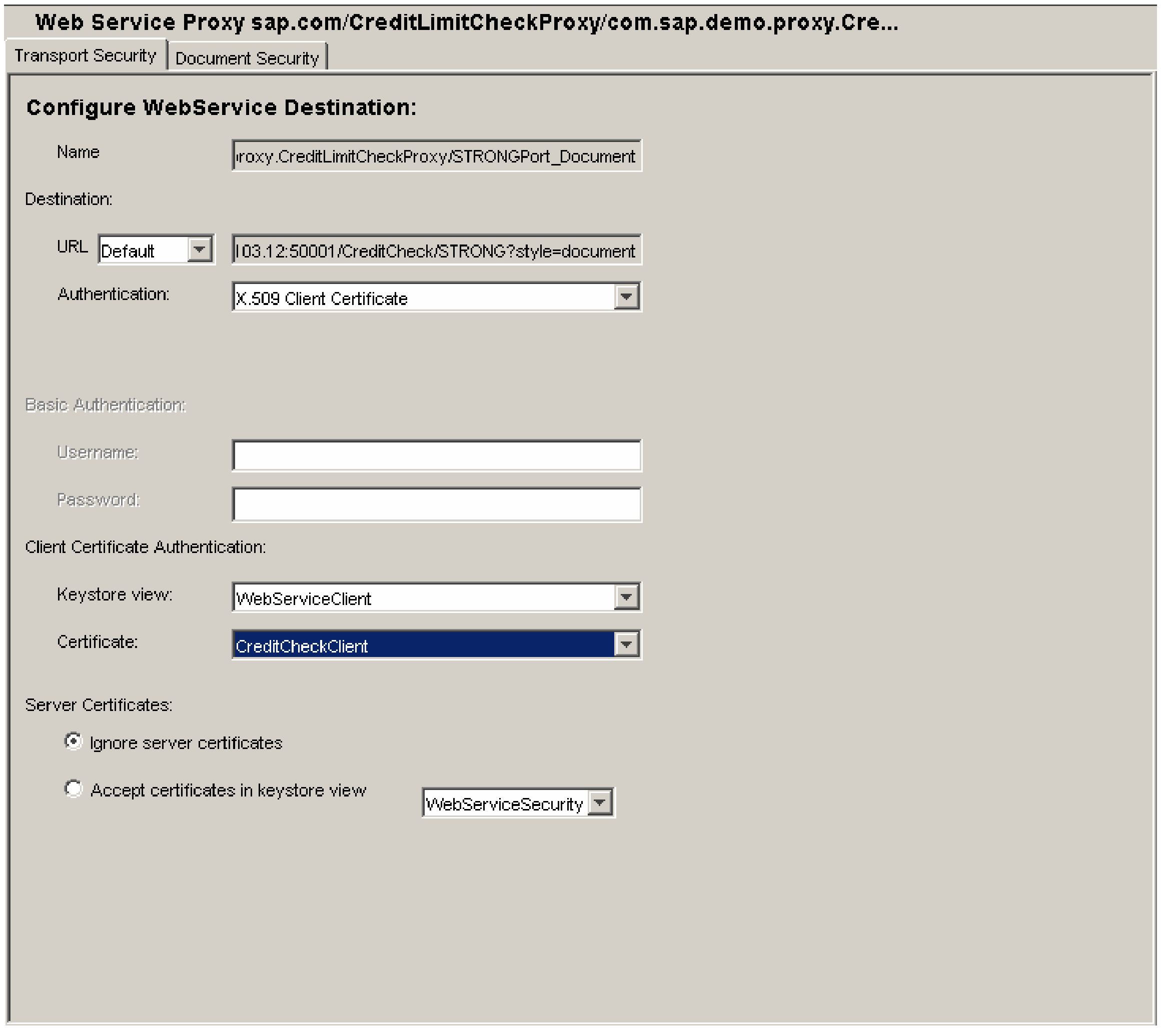
Groups: Everyone

Configuring the Web Service Client

1. In the Visual Administrator, select the node <*SID*> - *Server … - Services – Web Services Security*, where <SID> is the system ID of your client J2EE.
2. In the right window, activate the tabs *Runtime* (top) and *Security Administration* (bottom). A tree structure *Web Services* is displayed.
3. In this structure, select the node *Security Configuration – Web Service Clients – sap.com – CreditCheckProxy –*

*com.sap.demo.proxy.CreditLimitCheckProxy*STRONGPort_Document*. A window containing the security measures for this Web service client is displayed on the right. In this window, activate the *Transport Security* tab.

4. Under *Authentication*, choose the option **X.509 Client Certificate**. This enables you to enter additional logon data.
5. Under *Client Certificate Authentication, select the keystore view in which you stored the created certificate* (for example, WebServiceClient) and then select the relevant certificate (for example, CreditCheckClient).
6. Save your changes by choosing *Save* below the *Web Services* tree structure.



Web Service Proxy sap.com/CreditLimitCheckProxy/com.sap.demo.proxy.Cre...

Transport Security | Document Security

Configure WebService Destination:

Name: roxy.CreditLimitCheckProxy/STRONGPort_Document

Destination:

URL: Default | 03.12:50001/CreditCheck/STRONG?style=document

Authentication: X.509 Client Certificate

Basic Authentication:

Username:

Password:

Client Certificate Authentication:

Keystore view: WebServiceClient

Certificate: CreditCheckClient

Server Certificates:

[✓ Ignore server certificates]

Accept certificates in keystore view: WebServiceSecurity

Result

If the Web service client application *CreditLimitCheck* is executed, it logs on when the Web service *CreditCheck* is accessed by sending the specified certificate. If the system of the Web service provider is already configured appropriately, you can now test the client application. The following tutorial shows you how to configure the Web service for user authentication using a certificate: .

You can test the client application by calling the URL http://<host>:<port>/CreditLimitCheck in your browser. A simple screen for creating orders is displayed. After choosing the *Order* button, you are requested to enter the *Customer Identity Card Number.* If you press the *Check* button, the Web service CreditLimitPrivateLocal is called and the value you entered in the text field is passed. The result is then evaluated and displayed. Note that only data for the customer numbers 1001 to 1005 is displayed. For all other values, a message is displayed stating that no customer was found for that number.

Next Step:

Configuring the Web Service Client for Securing the Server Identity [Page 14]

Configuring the Web Service Client for Securing the Server Identity (Optional)

When a new SSL connection is opened, the server sends a certificate to the client for purposes of authentication. If the client is only to accept specific certificates, you must first import these certificates into the client and then define them as acceptable. By default, the Web service client accepts every certificate from the server.

Importing the server certificate is only necessary if your Web service client and the Web service to be consumed are not on the same J2EE. If this is not the case, you can skip the importing step and continue with the section *Configuring the Web Service Client for Checking the Server Identity* (see below).

Prerequisites

- You can access the certificate, which was exported as a crt file and is used by the server for authentication, via your file system.

Procedure

Importing the Server Certificate

1. Start the Visual Administrator and log on to the client J2EE.
2. Select the node <*SID*> - *Server … - Services – Key Storage* (where <SID> is the system ID of the client J2EE) and activate the tabs *Runtime* (top) and *Data* (bottom) in the right window.
3. Create a new keystore view by choosing the *Create View* button below the *Views* area. Give the new view a descriptive name - for example, *TrustedServers* - and confirm your entry. The new entry is displayed in the *Views* area.
4. Select your newly created keystore view.
5. In the *Entry* section (bottom right), choose *Load*. In the window that opens, select the certificate exported by the Web Service provider and choose *OK*. The certificate is now imported and is listed in the *Entries* field.

Configuring the Web Service Client for Checking the Server Identity

1. In the Visual Administrator, select the node **<SID> - Server … - Services – Web Services Security**, where <SID> is the system ID of your client J2EE.
2. In the right window, activate the tabs **Runtime** (top) and **Security Administration** (bottom). A tree structure *Web Services* is displayed.
3. In this structure, select the node *Security Configuration – Web Service Clients – sap.com – CreditLimitCheckProxy – com.sap.demo.proxy.CreditLimitCheckProxy*STRONGPort_Document.* A window containing the security measures for this Web service client is displayed on the right. In this window, activate the *Transport Security* tab.
4. In the **Server Certificates** area, activate the option *Accept certificates in keystore view* and then select the keystore view created in the last section from the list (for example, *TrustedServers*). If you skipped that section, because your client J2EE and server J2EE are identical, choose **service_ssl** here.
5. Save your changes by choosing *Save* below the Web service tree structure.

Result

The certificate sent by the server is now checked by the Web service client and only accepted if it is valid and is in the specified *keystore view*. Otherwise, the SSL connection is rejected and the Web service is not consumed.