

**Configuring Secure Socket Layer  
and Client-Certificate Authentication  
on SAS<sup>®</sup> 9.3 Enterprise BI Server  
Systems That Use Oracle WebLogic  
10.3**



---

## Table of Contents

---

<b>Overview .....</b>	<b>1</b>
<b>Configuring One-Way Secure Socket Layer.....</b>	<b>1</b>
Creating the Certificates and the Keystore That Are Required for One-Way SSL....	2
Installing and Configuring OpenSSL.....	2
Creating the Certificates and Keystores .....	2
Configuring One-Way SSL between an Administrator’s Browser and the Administration Server (AdminServer).....	3
Configuring the Administration Server .....	4
Importing Your CA Certificate to the Browser .....	6
Configuring One-Way SSL between the Administration Server and the Node Manager .....	7
Configuring One-Way SSL between a User’s Browser and a Managed Server (SASServer1).....	8
<b>Configuring Two-Way Secure Socket Layer .....</b>	<b>11</b>
<b>Configuring Client-Certificate Authentication.....</b>	<b>14</b>
<b>Glossary .....</b>	<b>17</b>



---

## Overview

---

This paper explains how to configure one-way Secure Socket Layer (SSL), two-way SSL, and client-certificate authentication on systems where you have SAS® 9.3 Enterprise BI Server installed and where you are using Oracle WebLogic 10.3 as your Java application server.

The document covers every aspect of these tasks, including the following:

- creating the keys, certificate requests, certificates, and keystores that are needed for the project. This paper explains how to create these objects using the OpenSSL toolkit and the Key and Certificate Management tool (keytool). These objects are created in such a way that you can easily replace the certificates that are documented in this paper with certificates from a third-party certificate authority.
- configuring WebLogic so that it can use SSL. This information is available in the WebLogic documentation. However, the information is included in this paper for convenience.
- configuring your SAS Enterprise BI system to use SSL.

**Note about authentication on enterprise BI systems:** By default, SAS enterprise BI systems are configured so that the users of Web applications are authenticated via SAS authentication. In this situation, the metadata server carries out the authentication. These systems can be converted to use Web authentication, in which WebLogic authenticates users directly. For example, WebLogic might authenticate users against information that is held in a Lightweight Directory Access Protocol (LDAP) directory. If you plan to configure one-way SSL or two-way SSL without client-certificate authentication, you can use either SAS or Web authentication. However, if you plan to configure two-way SSL with client-certificate authentication, you **must** configure Web authentication. The procedure for configuring Web authentication is covered in *Configuring Oracle WebLogic Server 10.3 for Web Authentication with SAS 9.3 Web Applications*. ([support.sas.com/resources/thirdpartysupport/v93/appservers/ConfiguringWLSWebAuth.pdf](http://support.sas.com/resources/thirdpartysupport/v93/appservers/ConfiguringWLSWebAuth.pdf)).

---

## Configuring One-Way Secure Socket Layer

---

*One-way SSL* is used to encrypt information that passes between a client application and a server. One-way SSL requires that the server provide a valid certificate to the client. However, the client is not required to reciprocate with a certificate to the server.

In the examples in this document, the *client* is usually the Web browser on which the user interface for the Web application is displayed. The *server* is WebLogic (or another Java application server), where the bulk of the application is actually running.

To configure one-way SSL properly, you need to configure SSL for the following connections:

- the connection between the administrator's browser and the administration server (AdminServer)
- the connection between the administration server and the node manager
- the connection between the users' browsers and the managed server (SASServer1)

These topics are addressed in the following sections, after a discussion about how to create the certificates and keystores that are required for one-way SSL.

## Creating the Certificates and the Keystore That Are Required for One-Way SSL

---

This section explains how to create the certificates and keystores that you need in order to configure one-way SSL. The certificates are created with the OpenSSL toolkit. The keystores are managed with the `keytool` utility, which ships with the Oracle Java Development Kit.

### Installing and Configuring OpenSSL

You can obtain OpenSSL on the Internet. If you plan to install the program in a Windows operating environment, you can find and use a binary distribution. Otherwise, you can download the source and build the program.

### Creating the Certificates and Keystores

The example in this section provides step-by-step instructions for creating the required certificates and keystore that are necessary for one-way SSL.

- First, the example illustrates how to create a certificate-authority (CA) certificate that you can use to sign a server-identity certificate.
- Then, the example illustrates how to create a server-identity certificate and a keystore to hold it.

To create the certificates and the keystore, follow these steps:

1. Create a CA private key by submitting the following command from a system prompt:

```
openssl genrsa -des3 -out myCAKey.pem 1024
```

After you submit this command, the system prompts you for a passphrase that is required in order to access the private key. For purposes of this example, this passphrase is specified as `CAPrivKeyPassword`. You designate a value for the passphrase at the prompt.

2. Create the CA certificate (a self-signed root certificate):

```
openssl req -new -key myCAKey.pem -x509 -days 1095 -out myCA.crt
```

When you are prompted for a passphrase for `myCAKey.pem`, enter the value for `CAPrivKeyPassword`.

For the fields in the distinguished name, you can use any values, except when you are prompted for a value for **Common Name**. For this field, enter the name that you want to appear in a user's list of trusted root certificates when the user imports the certificate into a browser.

3. Create an identity keystore:

```
keytool -genkeypair -alias WLServerCert -keyalg RSA -keystore serverids.jks  
-keypass IPrivKeyPassword -storepass IKeystorePassword
```

In this command:

- `WLServerCert` is just a recommended alias for the keystore entry that contains the certificate and private key. The same is true for other values that begin with `WL` that are shown in the upcoming steps.

- **Important:** The passwords for the private key and the keystore are, respectively, *IPrivKeyPassword* and *IKeystorePassword*. The letter *I*, which stands for *identity*, is there to distinguish these passwords from similar ones. If you configure only one-way SSL, these passwords can—and should—be different. If you plan to configure two-way SSL later, the passwords **must be the same**, as explained later in “Configuring Two-Way Secure Socket Layer.”

This command performs three tasks:

- It generates a pair of keys, one public and one private.
- It creates a self-signed certificate that contains the public key.
- It creates a keystore in which the certificate and private key are stored.

After you submit the command, you are prompted for the elements of a distinguished name. The value of the **Common Name (What is your first and last name?)** field **must** be the fully qualified host name of the machine on which the keystore is to be used.

4. Generate a request for a server-identity certificate:

```
keytool -certreq -alias WLServerCert -keystore serverids.jks -keypass
  IPrivKeyPassword -storepass IKeystorePassword -file WLCertReq.csr
```

5. Generate a server-identity certificate:

```
openssl x509 -req -days 365 -in WLCertReq.csr -CA myCA.crt -CAkey
  myCAKey.pem -CAcreateserial -out WLIdentity.crt
```

When you are prompted for the passphrase for *myCAKey.pem*, enter the value for *CAPrivKeyPassword*.

6. Import the CA certificate into the server-identity keystore:

```
keytool -importcert -alias WLRootCert -file myCA.crt -keystore
  serverids.jks -storepass IKeystorePassword
```

7. Import the server-identity certificate into its keystore:

```
keytool -importcert -alias WLServerCert -file WLIdentity.crt
  -keystore serverids.jks -keypass IPrivKeyPassword
  -storepass IKeystorePassword
```

8. Import the CA certificate into the Java standard trust keystore, which is named *cacerts* in the following example:

```
keytool -import -alias WLRootCert -file myCA.crt
  -keystore JDK-installation-directory\jre\lib\security\cacerts
```

When you are prompted for the passphrase for the keystore, enter the value **changeit**.

**Tip:** Record the password that you use in each place that a passphrase placeholder appears in the previous steps. You will need these passwords when you configure SSL.

## Configuring One-Way SSL between an Administrator’s Browser and the Administration Server (AdminServer)

---

To configure one-way SSL between an administrator’s browser and the administration server, you must change the configuration on the administration server and then import your CA certificate into your administrator’s browser.

## Configuring the Administration Server

To configure the administration server, follow these steps:

1. Copy the file `serverid.jks` to the directory `wlserver_10.3\server\lib` in your WebLogic installation. The full pathname might be similar to this:  
`C:\Oracle\Middleware\wlserver_10.3\server\lib\serverids.jks`. (You can put the keystores elsewhere, but this file is where your demonstration keystores are located.)
2. Log on to the WebLogic administration console.
3. In the administration console, select **SASDomain ► Environment ► Servers ► AdminServer(admin)** to display the **Settings for AdminServer** pane.
4. Click the **General** tab.

The screenshot shows the 'Settings for AdminServer' page in the WebLogic Administration Console. The 'General' tab is selected. The page includes a 'Save' button and a description: 'Use this page to configure general features of this server such as default network communications.' Below this is a 'View JNDI Tree' link and a table of server configuration details.

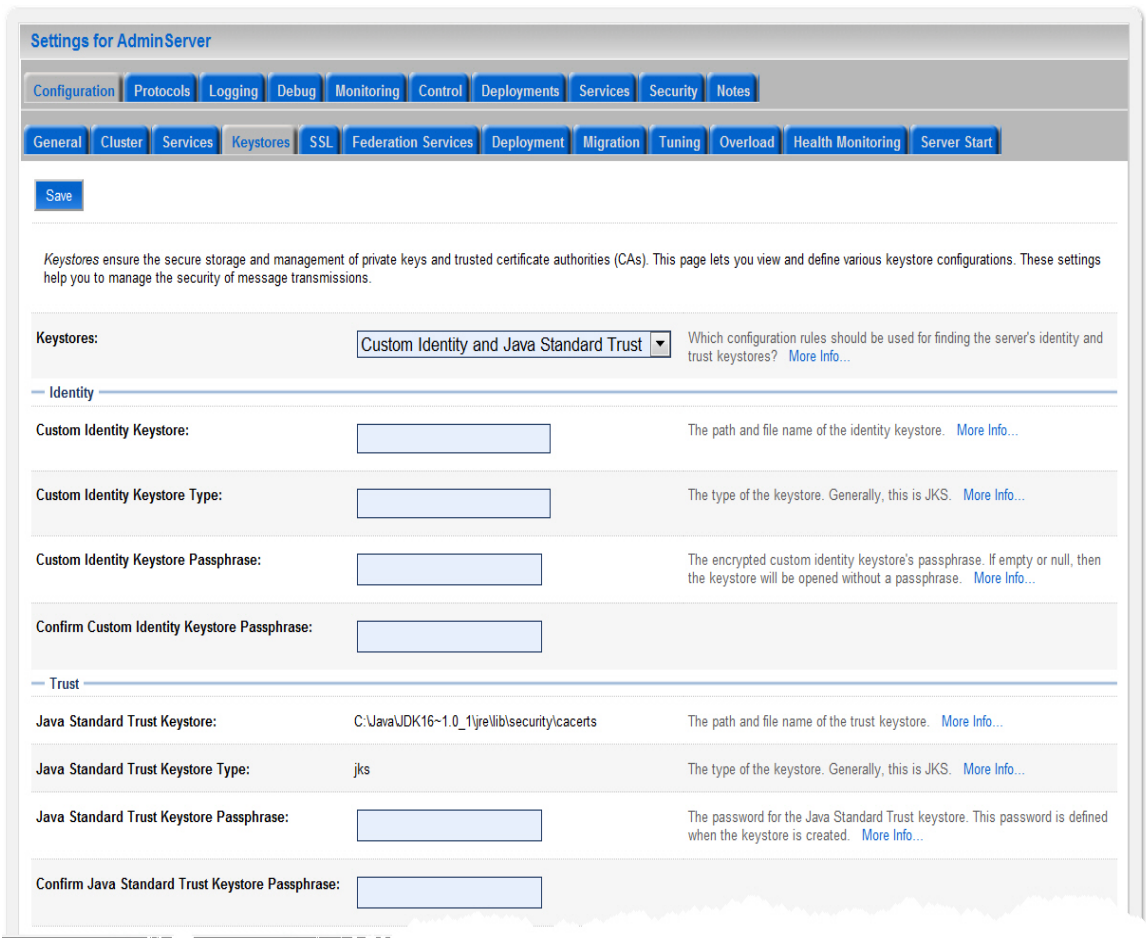
Property	Value	Description
Name:	AdminServer	An alphanumeric name for this server instance. <a href="#">More Info...</a>
Machine:	sasbi	The WebLogic Server host computer (machine) on which this server is meant to run. <a href="#">More Info...</a>
Cluster:	(Stand-Alone)	The cluster, or group of WebLogic Server instances, to which this server belongs. <a href="#">More Info...</a>
Listen Address:	<input type="text"/>	The IP address or DNS name this server uses to listen for incoming connections. <a href="#">More Info...</a>
<input checked="" type="checkbox"/> Listen Port Enabled		Specifies whether this server can be reached through the default plain-text (non-SSL) listen port. <a href="#">More Info...</a>
Listen Port:	<input type="text" value="7501"/>	The default TCP port that this server uses to listen for regular (non-SSL) incoming connections. <a href="#">More Info...</a>
<input type="checkbox"/> SSL Listen Port Enabled		Indicates whether the server can be reached through the default SSL listen port. <a href="#">More Info...</a>

5. On the **General** tab, make the following changes:
  - Select the check box for **SSL Listen Port Enabled**.
  - Change the value for **SSL Listen Port** to **7502**.

**Note:** For purposes of brevity, standard steps (for example, clicking the **Lock & Edit** button or activating your changes) are not included in these instructions. It is assumed that you understand when to perform these steps.



6. Click the **Keystores** tab.



7. Enter the following values for each of the items that are listed below:

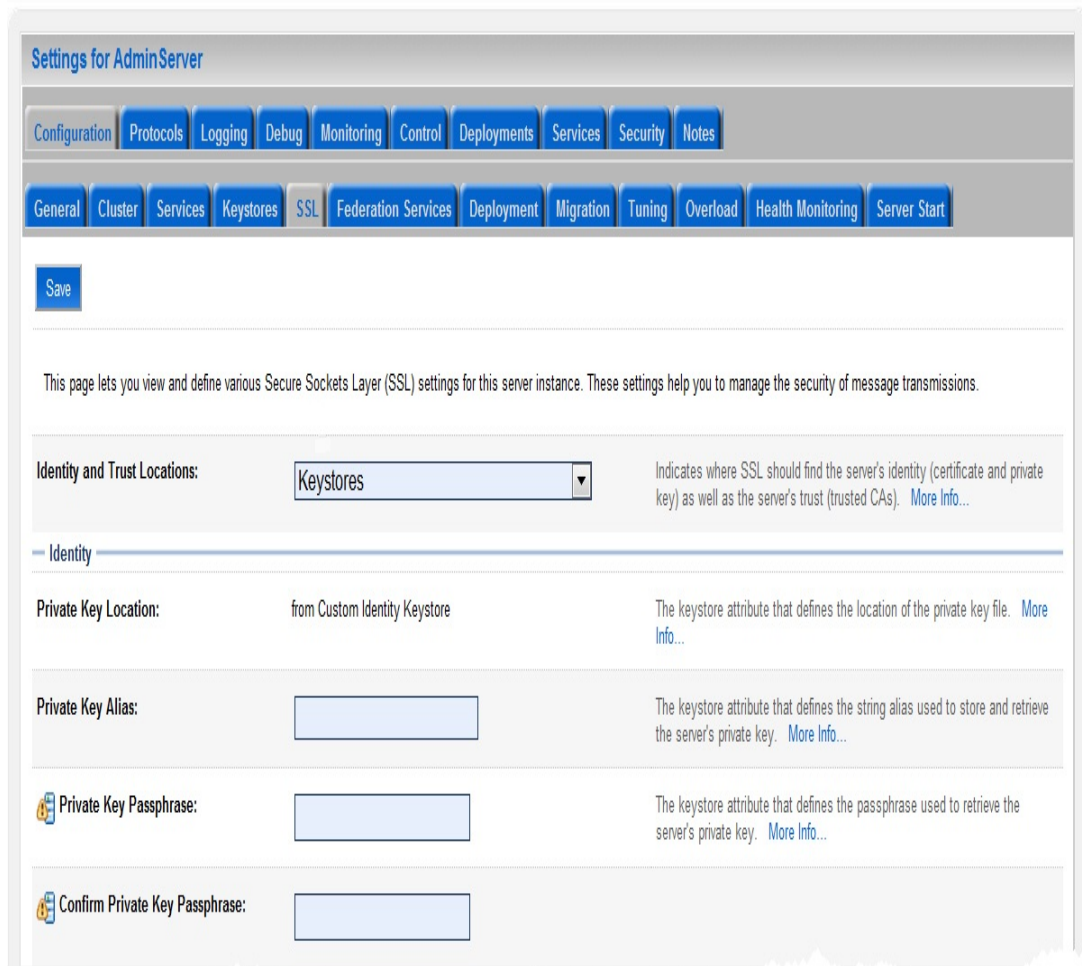
- **Keystores:** Select **Custom Identity and Java Standard Trust**. Selecting this value causes the pane to display additional text boxes, including **Custom Identity Keystore** text boxes.
- **Custom Identity Keystore:** Enter *path-to-server-IDs.jks*.
- **Custom Identity Keystore Type:** Enter **JKS**
- **Custom Identity Keystore Passphrase:** Enter the value for *IKeystorePassword*.

**Note:** You must enter this value twice. You enter it once in the **Custom Identity Keystore Passphrase** text box. Then you confirm it by entering the passphrase in the **Confirm Custom Identity Keystore Passphrase** text box.

- **Java Standard Trust Keystore Passphrase:** Enter the passphrase **changeit**, which is the passphrase for the keystore cacerts. If someone alters this passphrase, you have to use the new passphrase instead.

**Note:** You must enter this value twice. You enter it once in the **Java Standard Trust Keystore Passphrase** text box. Then you confirm it by entering the passphrase in the **Confirm Java Standard Trust Keystore Passphrase** text box.

8. Click the **SSL** tab.



9. On the **SSL** tab, enter the following values for each of the text boxes listed below:

- **Private Key Alias:** Enter the value **WLServerCert**.
- **Private Key Passphrase:** Enter the value for **IPrivKeyPassword**.

**Note:** You must enter this value twice. You enter it once in the **Private Key Passphrase** text box. Then you confirm it by entering the passphrase in the **Confirm Private Key Passphrase** text box.

10. Restart the administration server.

### Importing Your CA Certificate to the Browser

After you configure WebLogic with the information that it needs to obtain a server certificate, you must import the CA certificate for the CA that signed that certificate into your browser. This procedure varies a bit from browser to browser. If

you do not know how to import the certificate, search for instructions for your browser on the Internet. The following instructions are for Internet Explorer 7 under Windows:

1. Select **Tools ► Internet Options**.
2. In the Internet Options dialog box, click the **Content** tab. Then click the **Certificates** button.
3. In the Certificates dialog box, click the **Trusted Root Certification Authorities** tab. Then click the **Import** button, which activates the Certificate Import Wizard.
4. Use the wizard to import the certificate `myCA.crt`.

To test your work, navigate to `https://fully-qualified-hostname:7502/console`. If you reach the administration console's logon page without errors, the configuration is correct.

## Configuring One-Way SSL between the Administration Server and the Node Manager

To configure one-way SSL between the administration server and the node manager, follow these steps:

1. Stop the node manager.
2. In the administration console, navigate to **SASDomain ► Environment ► Machines ► machine-name** to display the **Settings for machine-name** pane.
3. Click the **Node Manager** tab, as shown in the following display:

Settings for machine-name

Configuration Monitoring Notes

General Node Manager Servers

Save

This page allows you to define the Node Manager configuration for this machine. To control a Managed Server from the console, Node Manager must be configured and running on the machine where the Managed Servers are installed.

The settings defined on this page are used to configure communication between the current domain and Node Manager instances that control Managed Servers. This page does not control the configuration of the Node Manager instances.

Type:  Returns the node manager type. [More Info...](#)

Listen Address:  The host name or IP address where Node Manager listens for connection requests. [More Info...](#)

Listen Port:  The port number where Node Manager listens for connection requests. [More Info...](#)

Node Manager Home:  Returns the nodemanager home directory that will be used to substitute for the shell command template. [More Info...](#)

Shell Command:  Returns the local command line to use when invoking SSH or RSH node manager functions. [More Info...](#)

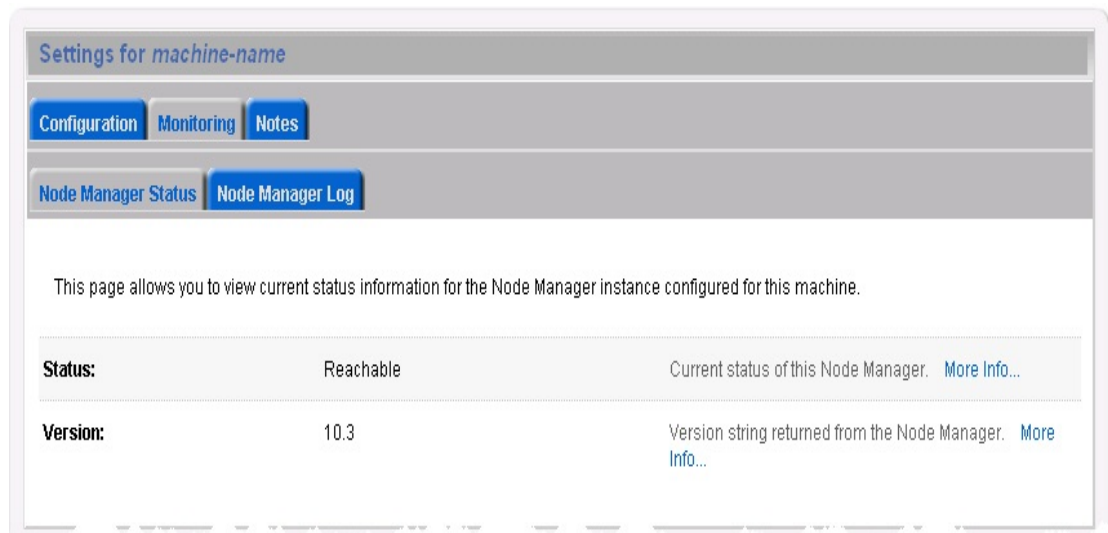
If the listen address is a short host name, change it to a fully qualified host name.

4. Edit the file `nodemanager.properties`, which is located in *SAS-configuration-directory*\Lev1\Web\SASDomain\nodemanager. Add the following lines to the end of the file, but replace the user-supplied values and file pathnames with your own values.

```
KeyStores=CustomIdentityAndJavaStandardTrust
CustomIdentityKeyStoreFileName=C:\:\Oracle\:\Middleware\:\wlserver_10.3
    \server\lib\serverids.jks
CustomIdentityKeyStorePassPhrase=IKeystorePassword
CustomIdentityKeyStoreType=JKS
CustomIdentityAlias=WLServerCert
CustomIdentityPrivateKeyPassPhrase=IPrivKeyPassword
```

WebLogic encrypts your passwords when you restart the node manager.

5. Restart the node manager.
6. Test your work as follows:
  - a. In the administration console, navigate to **SASDomain ► Environment ► Machines ► machine-name** to display the **Settings for machine-name** pane.
  - b. Click the **Monitoring** tab. If the node manager's status is **Reachable**, the configuration is correct.



## Configuring One-Way SSL between a User's Browser and a Managed Server (SASServer1)

This section explains how to configure one-way SSL between a user's browser and your managed server, SASServer1. After you make these changes, the managed server presents a server-identity certificate to all clients, including SAS Remote Services. Therefore, to complete the setup, you must configure Remote Services so that it can locate your trust keystore (`cacerts`), as explained in [Step 4 of this section](#).

To access your SAS applications via HTTPS, you also need to make some changes to two Java Virtual Machine (JVM) arguments that are specific to SAS and to some SAS metadata, as follows:

1. Configure the managed server (SASServer1) for one-way SSL. This task is simple. As long as the administration server and the managed server are running on the same host, you perform almost exactly the same steps on the managed server that you performed on the administration server previously in the section “[Configuring the Administration Server](#).” The only step that does not apply is Step 1. At this point, you have already copied your .jks file (the keystore) to an appropriate location. Instead of leaving the SSL listen port as the default value (7502), change it to 7002.

After that, import your CA certificate into users’ browsers, just as you imported that certificate into your administrators’ browsers earlier.

2. Edit the SAS metadata in order to indicate that you want to access your SAS Web applications with HTTPS. You need to make changes in the following applications and properties:
  - Configuration Manager plug-in
  - Server Manager plug-in
  - Foundations Services Manager plug-in
  - content mapping properties for the **SAS Folders** structure

You can make these changes in SAS® Management Console, as explained in the sections that follow.

### Configuration Manager

In the Configuration Manager plug-in, look at the properties for each application. If the Properties dialog box has a **Connection** tab, select that tab and make the following changes:

1. Change the **Communication Protocol** value to **HTTPS**.
2. Change the **Port Number** value to **7002**.

If no **Connection** tab is available, you can move on to the Server Manager plug-in.

### Server Manager

In the Server Manager plug-in, change the connection information for the SAS® Content Server:

1. Select **SAS Content Server**. When you make that selection, connection information for the server appears in the right pane.
2. Right-click the connection information and select **Properties**.
3. In the Properties dialog box, click the **Options** tab.
4. Change the value for **Application protocol** to **https** and the value for **Port number** to **7002**.

### Foundation Services Manager

In the Foundation Services Manager plug-in, edit the properties of the following services in order to specify a new URL for the SAS Content Server:

- **Remote Services**
- **SASBIPortlets4.3 Local Services**

(continued)

- **SASJSR168RemotePortlet4.3 Local Services**
- **SASPackageViewer4.3 Local Services**
- **SASPortal4.3 Local Services**
- **SASStoredProcess9.3 Local Services**
- **SASWebReportStudio4.3 Local Services**

For each set of services, follow these steps:


1. Expand the node for a set of services and, beneath that, expand the **Core** folder.
2. Right-click **Information Service** and select **Properties**.
3. Click the **Service Configuration** tab.
4. On the **Service Configuration** tab, click the **Configuration** button.
5. In the dialog box that opens, click the **Repositories** tab.
6. On the **Repositories** tab, select **WebDAV** and click **Edit**.
7. Change the **Port** value to **7002**, and select the **Secure** check box.

### Content Mapping for SAS Folders

Update the **Server** property on the **Content Mapping** tab for SAS Folders, as follows:

1. Select the **Folders** tab in SAS Management Console.
  2. Right-click **SAS Folders** and select **Properties**.
  3. In the SAS Folder Properties dialog box, click the **Content Mapping** tab.
  4. From the **Server** list, select **SAS Content Server**.
3. Edit the JVM arguments that are used to start the managed server, as follows:
- a. Select **SASDomain ► Environment ► Servers ► SASServer1** to display the **Settings for SASServer1** pane.
  - b. In the **Settings for SASServer1** pane, click the **Server Start** tab.
  - c. In the **Arguments** text area, make the following changes:

- Change the value of `-Dsas.auto.publish.port` to **7002**.

 Arguments:

```
Djavax.xml.soap.MessageFactory=com.sun.xml.messaging.saaj.soap.ver1_1.SOAPMessageFactory1_1Impl -
Dsas.auto.publish.port=7002 -
Djava.net.preferIPv4Stack=true -
Djava.net.preferIPv6Addresses=false -
Dmulticast_udp_ip_ttl=0 -Dmulticast.address=235.1.1.1 -
```

- Add the argument `-Dsas.auto.publish.protocol=https`.

4. Add the JVM options below to the wrapper.conf file (Windows) or the RemoteServices.sh file (UNIX operating systems).

**Under Windows:** Add the following JVM options to wrapper.conf, which is located in

**C:\SAS\Config\Lev1\Web\Applications\RemoteServices:**

```
wrapper.java.additional.13=-Djavax.net.ssl.trustStore="C:/Program
Files\Java\jdk1.6.0_24\jre\lib\security\cacerts"
wrapper.java.additional.14=-Djavax.net.ssl.trustStoreType=JKS
wrapper.java.additional.15=-Djavax.net.ssl.trustStorePassword=changeit
```

**Under UNIX:** In the RemoteServices.sh file, which is located in *SAS-configuration-directory/Lev1/Web/Applications/RemoteServices/*, add the options above to the line that begins with the argument **-Xms128m**.

These additional options are necessary because SAS Remote Services can also be a client of SASServer1. These additional options ensure that SAS Remote Services trusts the identity certificate that is sent by SASServer1.

5. Restart SAS Remote Services and SASServer1. You can test the one-way SSL by starting SASServer1 and invoking SAS® Web Report Studio. Invoke SAS Web Report Studio by using the URL **https://fully-qualified-hostname:7002/SASWebReportStudio**. If a logon page appears without any errors, your system is working correctly.

---

## Configuring Two-Way Secure Socket Layer

---

Two-way SSL provides not only the encryption capabilities of one-way SSL but also the ability to authenticate a user based on the contents of the user's digital certificate.

Configuring two-way SSL involves creating an identity certificate for the user and then configuring WebLogic so that it requests that certificate and determines whether it came from a trusted certificate authority. The steps in these procedures are as follows:

1. Create a client certificate that is signed by the CA that you defined earlier.
  - a. Create a new private key by submitting the following command:

```
openssl genrsa -des3 -out sasdemo.pem 1024
```

You will be prompted for a passphrase that is required in order to access the private key. In the following steps, **this passphrase is referred to by the placeholder value *sasdemoPassword***.

- b. Create a certificate request that is based on the new private key, as follows:

```
openssl req -new -key sasdemo.pem -out sasdemo.csr
```

When you are prompted, enter the passphrase (*sasdemoPassword*) for sasdemo.pem. For the fields in the distinguished name, you can use any values except when you are prompted for a common name. For the **Common Name** field, enter the user's ID (**sasdemo**).



- c. Create the client certificate and sign it by the CA that you created earlier by submitting this command:

```
openssl x509 -req -days 365 -in sasdemo.csr -CA myCA.crt -CAkey myCAKey.pem -CAcreateserial -out sasdemo.crt
```

When you are prompted for the passphrase for myCAKey.pem, enter the password *CAPrivKeyPassword*.

- d. Convert the client certificate to PKCS12 format so that you can import it into the user's browser.

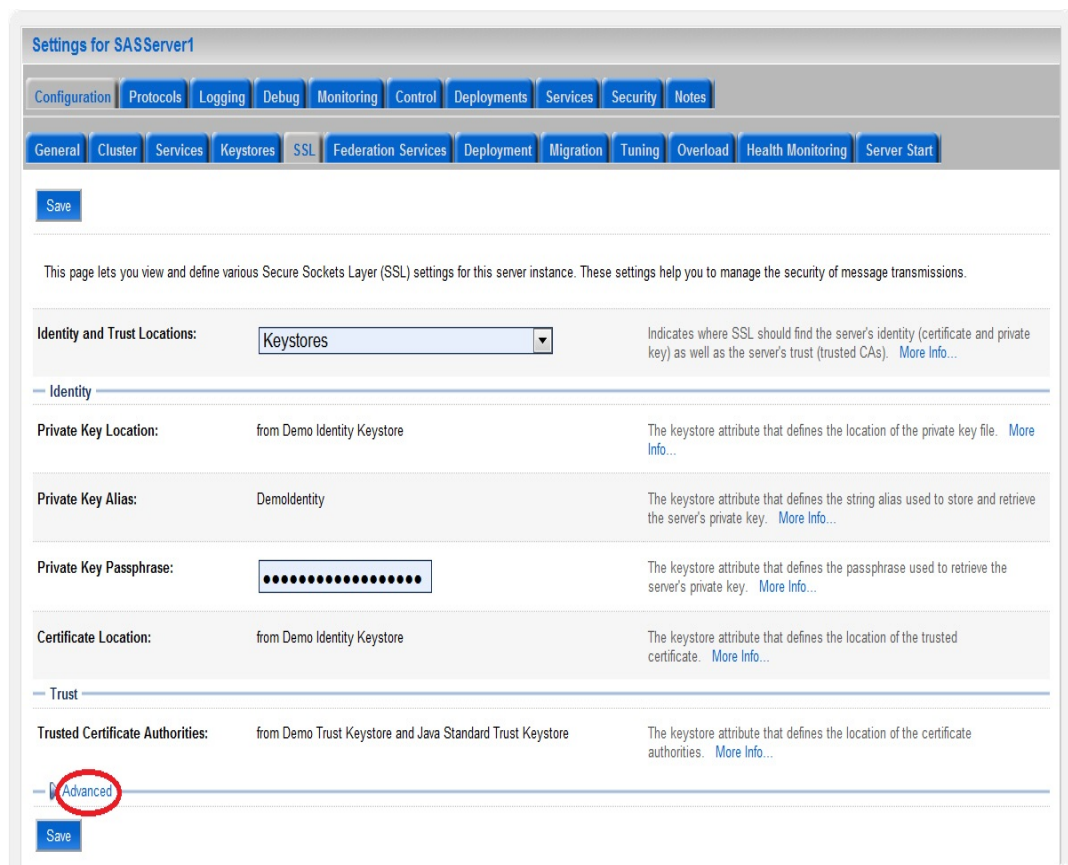
```
openssl pkcs12 -export -in sasdemo.crt -inkey sasdemo.pem -certfile myCA.crt -out sasdemo.p12
```

After you submit this command, you are prompted to provide an export password.

- 2. Import the client certificate sasdemo.p12 into each user's browser. You import the certificate similar to the way you imported myCA.crt earlier (step 6 in "Creating the Certificates and Keystores"). The only difference is that you should import the client certificate into the **Personal** certificate area. You are required to enter your export password during this procedure.

- 3. Configure WebLogic so that it requests and validates a client certificate, as follows:

- a. In the WebLogic administration console, select **SASDomain ► Environment ► Servers ► SASServer1** to display the **Settings for SASServer1** pane.
- b. Click the **SSL** tab, as shown in the following display:





- c. Click the **Advanced** link at the bottom of the pane to display the following additional text boxes:

The screenshot shows the 'Advanced' configuration pane with the following settings:

- Hostname Verification:** BEA Hostname Verifier (dropdown menu)
- Custom Hostname Verifier:** (empty text box)
- Export Key Lifespan:** 500 (text box)
- Use Server Certs:** (unchecked checkbox)
- Two Way Client Cert Behavior:** Client Certs Not Requested (dropdown menu)
- Cert Authenticator:** (empty text box)
- SSLRejection Logging Enabled:** (checked checkbox)
- Allow Unencrypted Null Cipher:** (unchecked checkbox)
- Inbound Certificate Validation:** Builtin SSL Validation Only (dropdown menu)
- Outbound Certificate Validation:** Builtin SSL Validation Only (dropdown menu)

A 'Save' button is located at the bottom left of the pane.

- d. Select **Client Certs Requested and Enforced** from the drop-down list for **Two Way Client Cert Behavior**.
- e. In the **Settings for SASServer1** pane, click the **Server Start** tab.

The screenshot shows the 'Server Start' configuration pane with the following settings:

- Java Home:** C:\Java\jdk1.6.0\_16
- Java Vendor:** Sun
- BEA Home:** C:\bea\wserver\_10.3
- Root Directory:** (empty text box)
- Class Path:** (empty text box)
- Arguments:**

```
-
Dcom.sun.xml.namespace.QName.useCompatibleSerialVersionUID=1
.0 -Dsas.server.name=Server -Xms768m -Xmx768m -
XX:PermSize=512m -XX:MaxPermSize=512m -Xss160k -
XX:NewSize=128m -XX:MaxNewSize=256m -
XX:+UseConcMarkSweepGC -XX:-UseTLAB -XX:+DisableExplicitGC
```

A 'Save' button is located at the top left of the pane.

- f. In the **Arguments** text box, scroll to the end of the current list of arguments and add the following arguments:

```
-Djavax.net.ssl.trustStore="C:\Program Files\Java\jdk1.6.0_24
                        \jre\lib\security\cacerts"
-Djavax.net.ssl.trustStoreType=JKS
-Djavax.net.ssl.trustStorePassword=changeit
-Djavax.net.ssl.keyStore="C:\Oracle\Middleware\wlserver_10.3
                        \server\lib\serverids.jks"
-Djavax.net.ssl.keyStoreType=JKS
-Djavax.net.ssl.keyStorePassword=IKeystorePassword
```

**Note:** Both the existing arguments and the new ones must be on a single line and must be separated by a space.

4. Configure SAS Remote Services, as follows.

**Note:** You also need to use the JVM options that are shown in the previous step when you start SAS Remote Services.

**Under Windows:** Add lines similar to the following to the wrapper.conf file, which is located in the directory **C:\SAS\Config\Lev1\Web\Applications\RemoteServices\**:

```
wrapper.java.additional.16=-Djavax.net.ssl.keyStore="C:\Oracle\ Middleware
                        \wlserver_10.3\server\lib\serverids.jks"
wrapper.java.additional.17=-Djavax.net.ssl.keyStoreType=JKS
wrapper.java.additional.18=-Djavax.net.ssl.keyStorePassword=IKeystorePassword
```

**Under UNIX:** In the RemoteServices.sh file, add the JVM options that are located in the directory **SAS-configuration-directory/Lev1/Web/Applications/RemoteServices/** to the line that begins with the argument **-Xms128m**.

5. Test the two-way SSL by restarting SAS Remote Services and SASServer1. Then try to run SAS Web Report Studio using the URL **https://fully-qualified-hostname:7002/SASWebReportStudio**. If you get to a logon page without any errors, your system is working correctly.

---

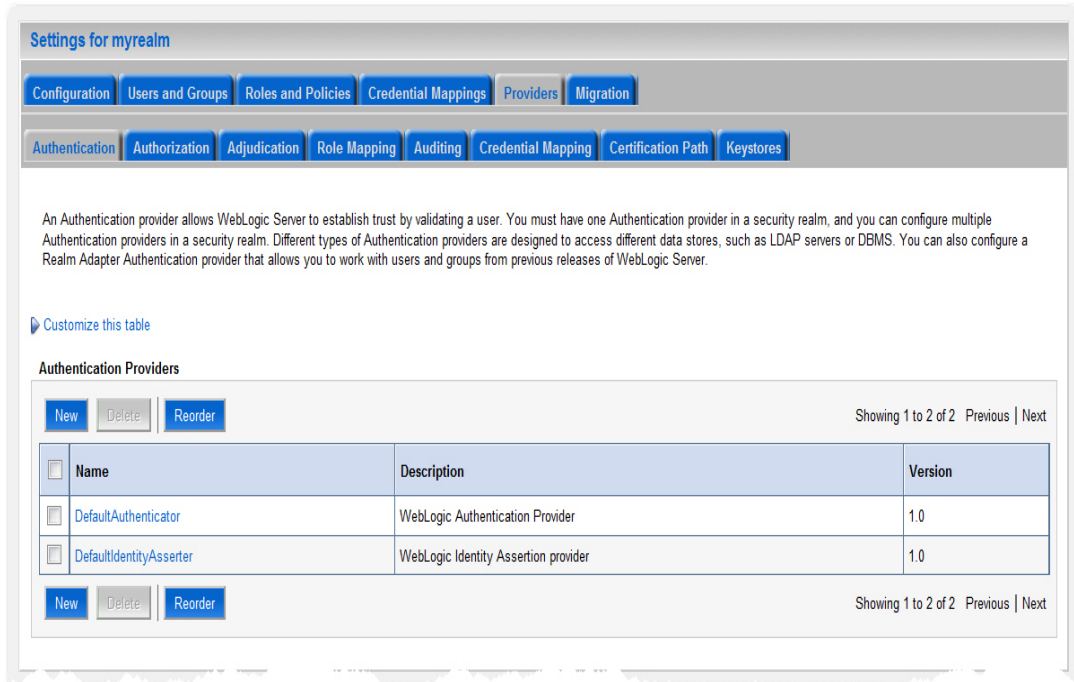
## Configuring Client-Certificate Authentication

---

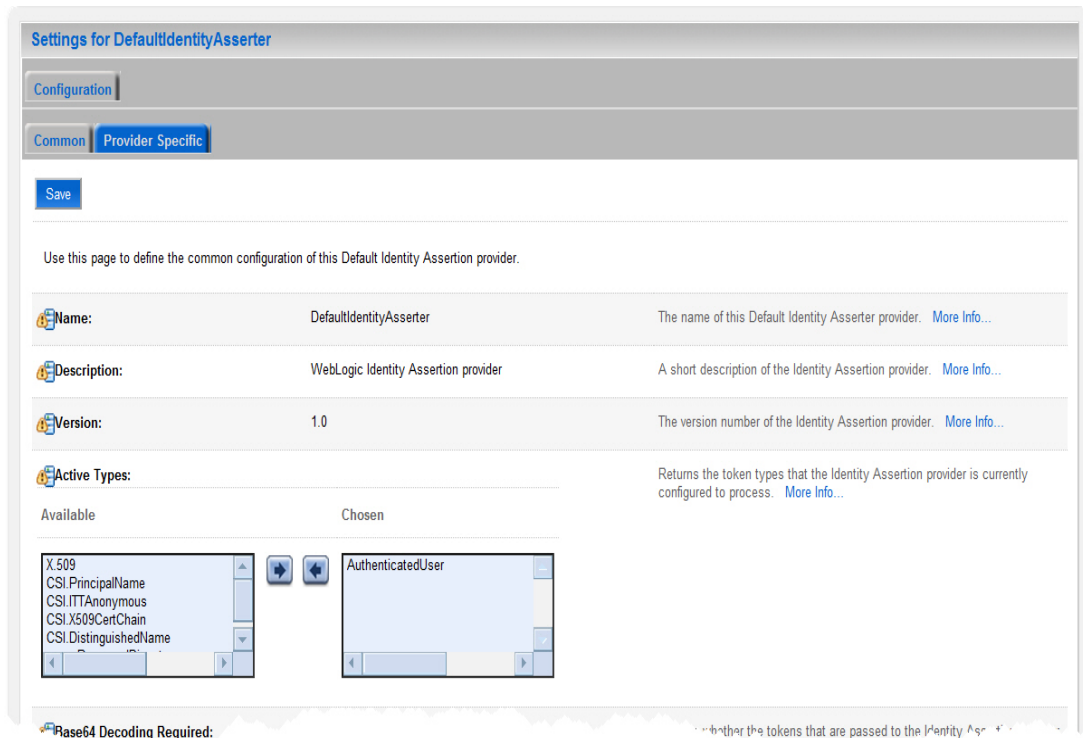
To complete the configuration for the client-certificate authentication, follow these steps:

1. Configure the WebLogic DefaultIdentityAsserter module so that it can search for and find the client's ID in an X509 certificate.
  - a. Log on to the WebLogic administration console.
  - b. Select **SASDomain ► Security Realms ► myrealm** to display the **Settings for myrealm** pane.

- c. Click the **Providers** tab.



- d. Click the **DefaultIdentityAsserter** link in the **Name** column to display the **Settings for DefaultIdentityAsserter** pane.
- e. Click the **Common** tab.



- f. In the **Active Types** list, move the item **X.509** from the **Available** list to the **Chosen** list.

- g. Click the **Provider Specific** tab.

- h. Set the following values:

- Enter a comma in the **Default User Name Mapper Attribute Delimiter** text box.
  - Select **CN** from the **Default User Name Mapper Attribute Type** list.
  - Select the check box **Use Default User Name Mapper**.
2. Configure the SAS Logon Manager so that it uses **client-certificate authentication** rather than **basic authentication**. When you configured Web authentication previously, you exploded the `sas.wip.apps9.3.ear` and the `sas.svcs.logon.war`, and you made changes to the SAS Logon Manager deployment descriptor file (`web.xml`). You need to use that same process again, but this time, follow these steps:
- a. Change the element `<auth-method>BASIC</auth-method>` to `<auth-method>CLIENT-CERT</auth-method>`.
  - b. Re-assemble the `.war` and `.ear` files.
  - c. Remove the current `sas.wip.apps9.3.ear` file and replace it with the new one.
3. Test the client-certificate authentication, as follows:
- a. Restart SAS Remote Services and your WebLogic servers.
  - b. Navigate to `fully-qualified-hostname:7002/SASWebReportStudio` in your browser. The system is configured correctly if this process bypasses your browser's logon dialog box and logs on to the application as the user that is named in your client certificate.

---

## Glossary

---

**certificate**

A digitally signed statement from one entity (a company or person) that says that the public key and other information from another entity have a certain value. When data is digitally signed, its authenticity and integrity can be validated by the signature.

**keystore**

A repository that holds digital certificates and private keys that are used to identify client or server machines. Because of the security information held in a keystore, it is password protected.

**keytool**

A tool that stores keys and certificates in a keystore and which you can use to manage keystores.

**one-way SSL**

A Secure Socket Layer (SSL) protocol that is used to encrypt information that passes between a client application and a server. One-way SSL requires that the server provide a certificate to the client. However, the client is not required to reciprocate with a certificate to the server.

**OpenSSL tool**

An open-source toolkit that implements both the Secure Socket Layer (SSL) and the Transport Layer Security (TLS) protocols. The tool also contains a robust cryptography library for security purposes.

**passphrase**

A string of words or other text that are used to access a computer system. Passphrases are similar to passwords; however, they are usually longer, so as to provide additional security.

**private key**

A number that is known only to its owner. The owner uses the private key to read (decrypt) an encrypted message.

**Secure Socket Layer (SSL)**

A protocol developed by Netscape Communications that provides network security and privacy. SSL, which provides a high level of security, uses encryption algorithms RC2, RC4, DES, TripleDES, and AES.

**two-way SSL**

A Secure Socket Layer configuration that provides the encryption capabilities of one-way SSL and enables authentication of a user based on the contents of the user's digital certificate. In two-way SSL, the server is required to present a certificate to the client, and the client is required to reciprocate with a certificate.

