Configuring IBM WebSphere Application Server 7 for Secure Sockets Layer and Client-Certificate Authentication on SAS® 9.3 Enterprise BI Server Web Applications

Table of Contents

Overview

This document explains how to configure one-way Secure Socket Layer (SSL), two-way SSL, and client-certificate authentication with the IBM WebSphere application server for use with SAS® 9.3 web applications. The document explains the following tasks in detail:

- configuring the system for one-way SSL
- configuring your system for two-way SSL
- configuring client-certificate authentication

Note: To configure client-certificate authentication, your system must first be configured for basic web authentication. The procedure for configuring web authentication is covered in *Configuring IBM WebSphere Application Server 7.0 for Web Authentication with SAS® 9.3 Web Applications*. For more details, see the section "Configuring WebSphere for Web Authentication," later in this document. If you are configuring one-way SSL only, web authentication is optional. You can use the default SAS host authentication instead.

Configuring One-Way Secure Socket Layer

One-way SSL authentication requires that a server machine provide a valid certificate to a client machine so that the client can verify the identity of the server. Because the authentication is only one way, the client is not required to provide a certificate to the server.

In this document, the *client* is the web browser on which the web application is displayed. The *server* is WebSphere, from which the application actually runs.

The basic steps for setting up one-way SSL are as follows:

1. Create a server-identify certificate, sign it with a Certification-Authority (CA) certificate, and add it to a Java keystore.
2. Import the server's certificate into your browser's trusted signer's area.
3. Configure WebSphere with the appropriate password so that it can access the keystore.
4. Use SAS® Management Console to change the connection information for the SAS Logon Manager and the set of SAS web applications that you want to access via Hyper Text Transfer Protocol Secure (HTTPS). These web applications might include the SAS® Content Server and WebDAV URLs.

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. Keystores and certificates are created with the OpenSSL toolkit. The keystores are managed with the IBM iKeyman utility and the SSL management capabilities in the WebSphere administration console. For more details, see "Creating keystores" in the *WebSphere Application Server v7.0 Security Guide*.

Note: The sample commands that are used in this document represent one of many ways you can accomplish the tasks that are illustrated.

Installing and Configuring the OpenSSL Toolkit

You can obtain OpenSSL from 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.

Before you create a self-signed certificate in the sections that follow, you need to perform the following steps:

1. Create a directory (**`C:\`*`directory`***) in which to store all of the certificates and keystores. (**Example: `C:\SSL`**)
2. In **`C:\`*`directory`***, create the following subdirectories:
   - **`C:\`*`directory`*`\demoCA`**
   - **`C:\`*`directory`*`\keys`**
   - **`C:\`*`directory`*`\certs`**
   - **`C:\`*`directory`*`\requests`**
3. In **`C:\directory\demoCA`**, create the following subdirectory:

   `C:\`*`directory`*`\demoCA\newcerts`
4. Copy ***`OPENSSL-directory`*`\bin\PEM\demoCA\serial`** to **`C:\`*`directory`*`\demoCA`**, as shown in the following example:

   ```
   C:\SSL\demoCA>copy c:\OpenSSL\bin\PEM\demoCA\serial
   ```
5. Create an empty text file called index.txt in the **`demoCA`** subdirectory, as shown below:

   `C:\`*`directory`*`\demoCA\index.txt`

Creating the Certificates and Keystores

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

- It creates a CA certificate that you can use to sign a server-identity certificate.
- It creates a server-identity certificate and a keystore to hold it.
- It imports the CA certificate into a trusted signer's keystore.

The following example assumes that certificate-related files and keystores are placed in a location that is relative to the WebSphere server. However, that does not necessarily have to be the case. In addition, the commands that are used in this example represent one of many ways to accomplish the tasks. For example, rather than submitting each entry at a prompt, OpenSSL enables you to specify a distinguished name (DN) on the command line. For more details, see the "OpenSSL Cryptography and SSL/TLS Toolkit" website (available at **`www.openssl.org/`**) and the Oracle document "keytool – Key and Certificate Management Tool" (available at (**`docs.oracle.com/javase/6/docs/technotes/tools/solaris/keytool.html`**).

For simplicity, the examples use the same password for all certificates and keystores, but they can be defined as unique passwords as well.

Note: In the following sections, paths and filenames that are used in commands are relative to the directory that you created previously in step 1 of the section "Installing and Configuring the OpenSSL Toolkit."

To create the certificates and the keystore:

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

   ```
   openssl genrsa -des3 -out keys/CA-key-name.pem 1024
   ```

   In this syntax, *CA-key-name* specifies the name of the CA private key.

   Note: The private key should be stored in a secure location in a production environment.

   ```
   openssl genrsa -des3 -out keys/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. Enter the passphrase when you are prompted. In subsequent steps, the passphrase is represented by the placeholder *passphrase.*

2. Generate the CA root certificate by using the following command syntax:

   ```
   openssl req –new –key keys/CA-key-name.pem –x509 –days 1095
               –out certs/CA-certificate-name.crt
   ```

   In this command, *CA-certificate-name* specifies the name of the CA root certificate.

   For testing purposes, this example creates a CA certificate. However, you can also use a self-signed certificate or it can come from a Certification Authority. Fill in the fields for the distinguished name (DN) when you are prompted, and provide the same passphrase that you specified in step 1.

   Note: For the fields in the DN, 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 the user's list of trusted root certificates when the user imports the certificate into a browser.

   You can also just add all of the information in the command at one time, as follows, thereby avoiding the prompts:

   ```
   openssl req -new -key keys/CA-key-name.pem -x509 -days 1024
               -subj /C=US/OU=TSD/O=SAS/ST=NC/L=Cary/CN=CA-name
               -out certs/CA-certificate-name.crt
   ```

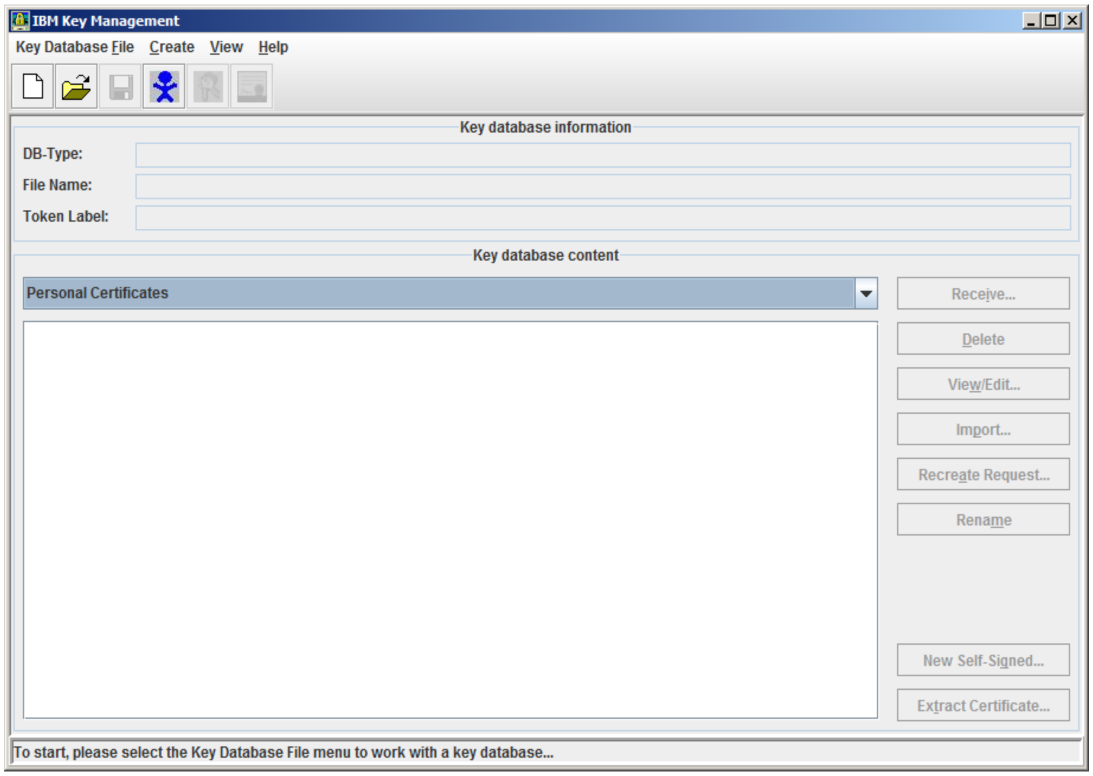
   In this instance, *`CA-name`* specifies a name that you want to use to reference the Certification Authority.

   Example:

   ```
   openssl req –new –key keys/myCAkey.pem –x509 –days 1024 –subj
        /C=US/OU=TSD/O=SAS/ST=NC/L=Cary/CN=sasbi -out certs/myCA.crt
   ```

3. Generate the server-identity keystore, as follows:

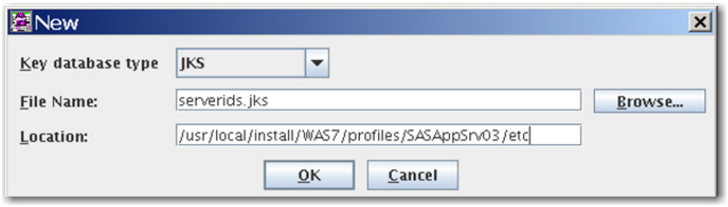
   a. Run the iKeyman utility, which is located in ***`WebSphere-home-directory`*****`/bin/.`**



To start, please select the Key Database File menu to work with a key database...

   b. From the iKeyman panel menu, select **Key Database File ► New**.

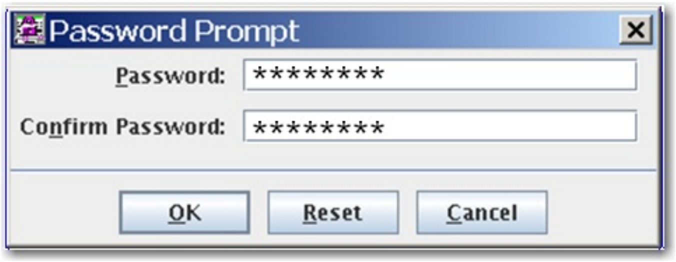
   c. In the New dialog box, select **`JKS`** as the value from the **`Key database type`** list. Then enter values for **`File Name`** (for example, serverids.jks) and **`Location`**.



Key database type: JKS
File Name: serverids.jks
Location: /usr/local/install/WAS7/profiles/SASAppSrv03/etc

   d. Click **`OK`**. This action opens the Password Prompt dialog box.

   e. In the Password Prompt dialog box, enter a password for the keystore and confirm it.



Password: ********
Confirm Password: ********

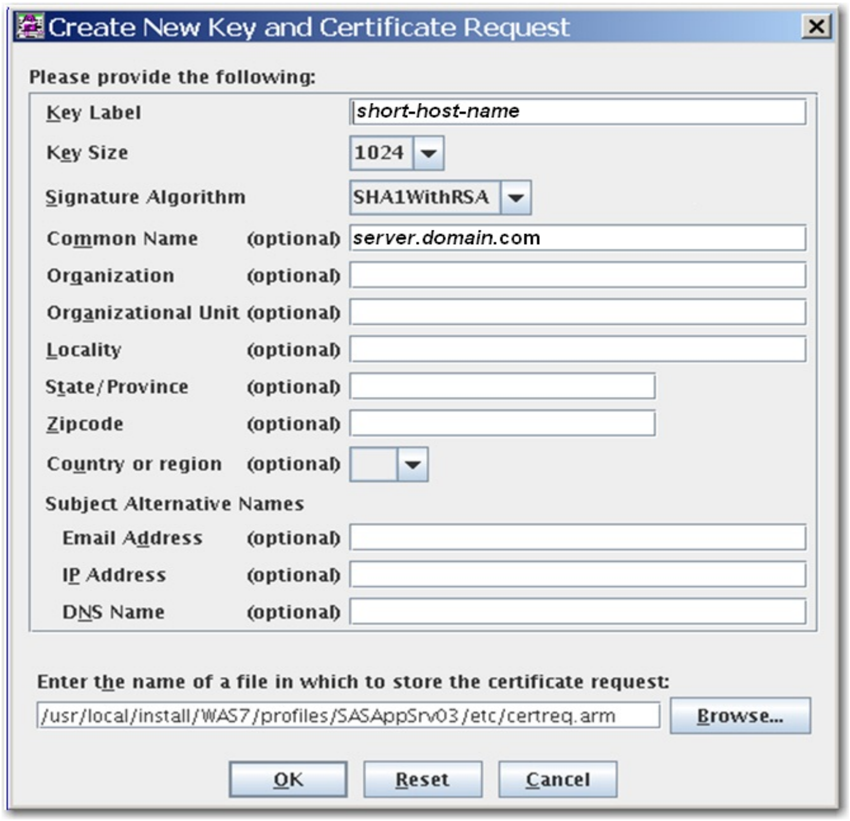
   f. Click **`OK`** to close the dialog box.

4. Generate the server-identity request file, which contains the server public key and the server identity.

   a. From the iKeyman panel, press Ctrl+R or click the short-cut icon for **`Create a new certificate request`**. This action opens the Create a New Key and Certificate Request dialog box.

   b. For the text boxes in the dialog box, specify the following:

      - **`Key Label`:** Specify the short host name for the ARM file in which the certificate request is to be stored.
      - **`Common Name`:** Specify the common name (CN), which should be the fully qualified host name.
      - **`DNS Name`:** Specify the complete DN. **Note:** This text box is optional.



Please provide the following:

Key Label		short-host-name
Key Size		1024
Signature Algorithm		SHA1WithRSA
Common Name	(optional)	server.domain.com
Organization	(optional)	
Organizational Unit	(optional)	
Locality	(optional)	
State/Province	(optional)	
Zipcode	(optional)	
Country or region	(optional)	
Subject Alternative Names		
Email Address	(optional)	
IP Address	(optional)	
DNS Name	(optional)	

Enter the name of a file in which to store the certificate request:

/usr/local/install/WAS7/profiles/SASAppSrv03/etc/certreq.arm

In addition, at the bottom of this dialog box, verify the filename for where the certificate request should be stored.

5. Click **`OK`** to close the dialog box.

6. Generate the server-identity certificate by submitting a command with the following syntax from a system prompt:

```
openssl x509 -req -days 40 -in requests/certificate-signing-request-
filename.csr -CA certs/CA-certificate-name.crt -CAkey keys/CA-key-name.pem
-CAcreateserial -out certs/server-certificate-name.crt
```

This certificate contains the server public key, the name of the machine on which JBoss runs, and the CA identity. It is also signed with the CA private key.

Example:

```
openssl x509 -req -days 40 -in requests/WebSphere.csr -CA certs/myCA.crt
             -CAkey keys/myCAkey.pem -CAcreateserial
             -out certs/WebSphere.crt
```

This certificate contains the server public key, the server identity, and the CA identity. This certificate is signed with the CA private key.

7. Import the CA root certificate (myCA.crt) into the Signer Certificates area of the server-identity keystore (serverids.jks).
   a. In the iKeyman utility, select **`Signer Certificates`** from the **`Key Database Content`** drop-down list.
   b. Click the **`Add`** button.
   c. Specify the CA certificate filename (*CA-certificate-name*.crt).
   d. When the utility prompts you, specify **`myCA`** as the **`Label`** value.
   e. Click **`OK`**. The certificate should appear now in the list of **`Signer Certificates`** in the iKeyman window.
8. Specify the signed server-identity certificate (*host-name*Signed.arm), as follows:
   a. In iKeyman, specify **`Personal Certificate`** from the **`Key Database Content`** drop-down list.
   b. Click the **`Receive`** button.
   c. Specify the signed server-identity certificate (*host-name*Signed.crt).
   d. Click **`OK`**. The certificate should appear now in the list of **`Personal Certificates`** in the iKeyman window.
9. Generate the trusted-CA-certificates keystore, as follows:
   a. From the iKeyman menu, select **Key Database ► New**.
   b. In the New dialog box, select **`JKS`** as the value for **`Key database type`**. Then specify values for **`File Name`** and **`Location`**.
   c. Click **`OK`**.
   d. When the utility prompts you for a password, enter a password for this keystore.
10. Import the CA root certificate (myCA.crt) into the trusted keystore (trustedca.jks).
    a. In the iKeyman utility, select **`Signer Certificates`** from the **`Key Database Content`** drop-down list.
    b. Click the **`Add`** button.
    c. Specify the CA certificate filename (*CA-certificate-name*.crt).
    d. When the utility prompts you, specify **`myCA`** as the **`Label`** value.
    e. Click **`OK`**. The certificate should appear now in the list of **`Signer Certificates`** in the iKeyman window.

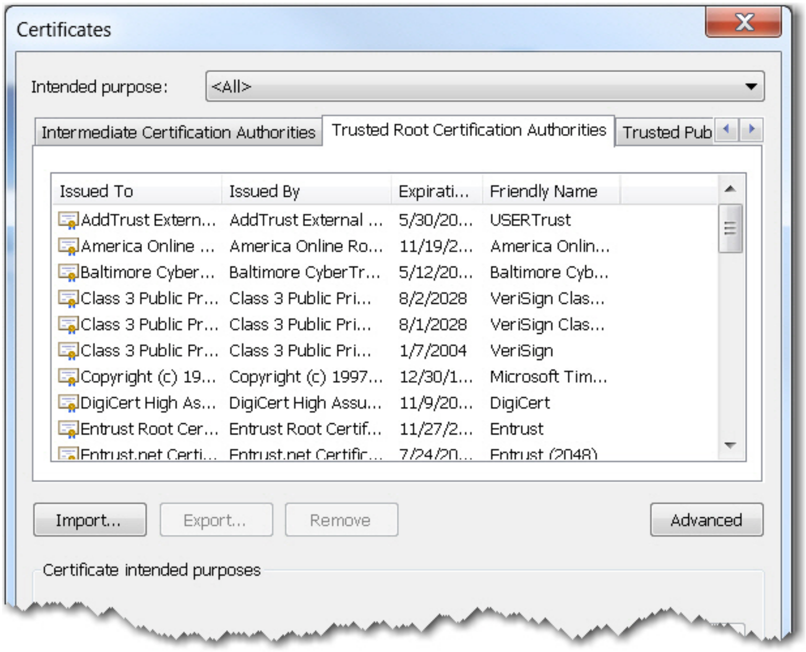
You now have the certificates and keystores that you need to configure one-way SSL.

Importing the CA Root Certificate into Your Browser

You need to import the CA root certificate (myCA.crt) that you created in the last section into your browser so that the browser trusts the server's certificate. Copy the certificate to the machine where you access the browser.

To import this certificate into Internet Explorer:

1. Select **Tools** ► **Internet options** ► **Content** ► **Certificates** ► **Trusted Root Certification Authorities**. This selection opens the Certificates dialog box.

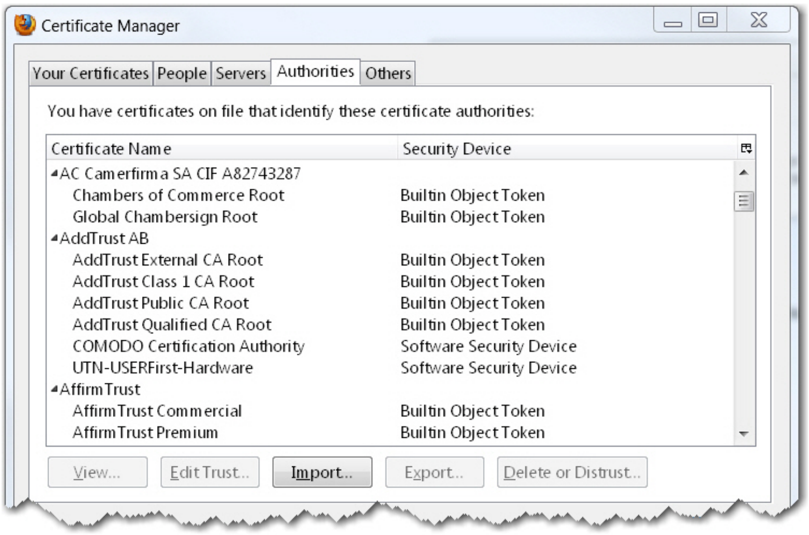

Certificates
Intended purpose: <All>
Intermediate Certification Authorities
Trusted Root Certification Authorities
Trusted Pub
Import...
Export...
Remove
Advanced
Certificate intended purposes

Issued To	Issued By	Expirati...	Friendly Name
AddTrust Extern...	AddTrust External ...	5/30/20...	USERTrust
America Online ...	America Online Ro...	11/19/2...	America Onlin...
Baltimore Cyber...	Baltimore CyberTr...	5/12/20...	Baltimore Cyb...
Class 3 Public Pr...	Class 3 Public Pri...	8/2/2028	VeriSign Clas...
Class 3 Public Pr...	Class 3 Public Pri...	8/1/2028	VeriSign Clas...
Class 3 Public Pr...	Class 3 Public Pri...	1/7/2004	VeriSign
Copyright (c) 19...	Copyright (c) 1997...	12/30/1...	Microsoft Tim...
DigiCert High As...	DigiCert High Assu...	11/9/20...	DigiCert
Entrust Root Cer...	Entrust Root Certif...	11/27/2...	Entrust
Entrust.net Certi...	Entrust.net Certific...	7/24/20...	Entrust (2048)

2. Click the **`Import`** button to start the Certificate Import Wizard, and follow the wizard's instructions. You need to supply the path to the certificate file. For all other values, accept the default values.

To import this certificate into Mozilla Firefox:

1. Select **Tools** ► **Options** ► **View Certificates** ► **Authorities**.



You have certificates on file that identify these certificate authorities:

Certificate Name	Security Device
AC Camerfirma SA CIF A82743287	
Chambers of Commerce Root	Builtin Object Token
Global Chambersign Root	Builtin Object Token
AddTrust AB	
AddTrust External CA Root	Builtin Object Token
AddTrust Class 1 CA Root	Builtin Object Token
AddTrust Public CA Root	Builtin Object Token
AddTrust Qualified CA Root	Builtin Object Token
COMODO Certification Authority	Software Security Device
UTN-USERFirst-Hardware	Software Security Device
AffirmTrust	
AffirmTrust Commercial	Builtin Object Token
AffirmTrust Premium	Builtin Object Token

2. Click the `Import` button to start the Certificate Import Wizard, and follow the wizard's instructions. You need to supply the path to the certificate file. For all other values, accept the default values.

Configuring the WebSphere Application Server for One-Way SSL

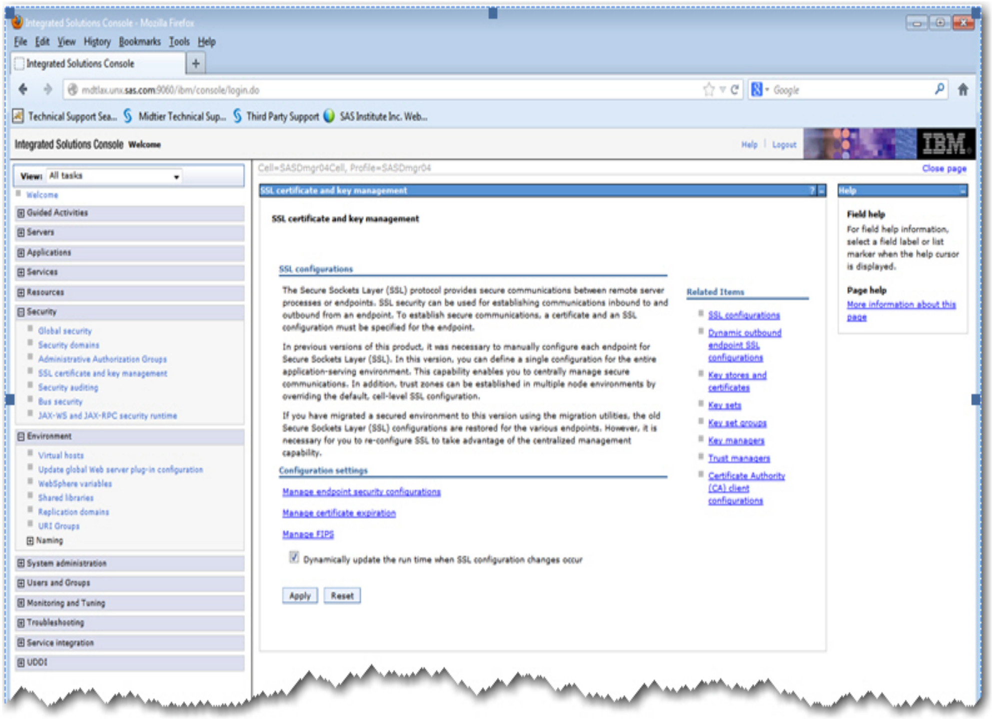
At the cell and node level, changes in SSL configuration can break communication between the cell and the node if they use the SSL connection when WebSphere administrative security is enabled. Typically, problems show up during node synchronization. **Therefore, it is highly recommended that you disable administrative security when you update the SSL configuration.** To do this, select **Security ► Global Security**, and then deselect the **`Enable administrative security`** check box. **Re-enable the administrative security after all SSL configuration changes are saved and synchronized between the cell and the node.** For more information about enabling or disabling WebSphere administrative security, see "Enabling administrative security" (section 2.2) and "Disabling administration security" (section 2.3) in the *WebSphere Application Server V7.0 Security Guide*. (`www.redbooks.ibm.com/redbooks/pdfs/sg247660.pdf`)

To configure WebSphere for one-way SSL:

All of the following steps occur in the WebSphere administrative console.

1. In the console, import the signer's certificate (root CA) into the node-level keystore, as follows:

   a. Select **SSL certificate and key management ► SSL configurations ► NodeDefaultSSLSettings ► Key stores and certificates ► NodeDefaultKeyStore ► Signer certificates**.


Integrated Solutions Console - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Integrated Solutions Console
mdtlax.unx.sas.com:9060/ibm/console/login.do
Technical Support Sea... Midtier Technical Sup... Third Party Support SAS Institute Inc. Web...
Integrated Solutions Console Welcome
Help | Logout
IBM
View: All tasks
Welcome
Guided Activities
Servers
Applications
Services
Resources
Security
Global security
Security domains
Administrative Authorization Groups
SSL certificate and key management
Security auditing
Bus security
JAX-WS and JAX-RPC security runtime
Environment
Virtual hosts
Update global Web server plug-in configuration
WebSphere variables
Shared libraries
Replication domains
URI Groups
Naming
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Service integration
UDDI
Cell=SASDmgr04Cell, Profile=SASDmgr04
Close page
SSL certificate and key management
Help
Related Items
SSL configurations
Dynamic outbound endpoint SSL configurations
Key stores and certificates
Key sets
Key set groups
Key managers
Trust managers
Certificate Authority (CA) client configurations
Apply
Reset

SSL certificate and key management

SSL configurations

The Secure Sockets Layer (SSL) protocol provides secure communications between remote server processes or endpoints. SSL security can be used for establishing communications inbound to and outbound from an endpoint. To establish secure communications, a certificate and an SSL configuration must be specified for the endpoint.

In previous versions of this product, it was necessary to manually configure each endpoint for Secure Sockets Layer (SSL). In this version, you can define a single configuration for the entire application-serving environment. This capability enables you to centrally manage secure communications. In addition, trust zones can be established in multiple node environments by overriding the default, cell-level SSL configuration.

If you have migrated a secured environment to this version using the migration utilities, the old Secure Sockets Layer (SSL) configurations are restored for the various endpoints. However, it is necessary for you to re-configure SSL to take advantage of the centralized management capability.

Configuration settings

Manage endpoint security configurations

Manage certificate expiration

Manage FIPS

[✓] Dynamically update the run time when SSL configuration changes occur

Field help

For field help information, select a field label or list marker when the help cursor is displayed.

Page help

More information about this page

   b. Click **Add** to display additional text boxes.

   c. In the **`Alias`** text box, enter the **`Label`** value that you specified previously in step 7(d) of the section "Creating the Certificates and Keystores."

d. In the **`Filename`** text box, enter the path to the CA certificate (myCA.crt) that you specified in step 2 of "Creating the Certificates and Keystores."

e. Click **OK**.

f. Review your entries and save the changes when you are prompted.

2. Import the signed server certificate from the serverids.jks keystore and the signer's certificate (root CA) into the node-level keystore, as follows:

   a. Select **Security ► SSL certificate and key management ► SSL configurations ► NodeDefaultSSLSettings ► Key stores and certificates ► NodeDefaultKeyStore ► Personal certificates ► Import**.

   b. Select **`Key store file`**.

   c. In the **`Key file name`** text box, enter the path to the keystore that you created in step 3(c) of the section "Creating the Certificates and Keystores."

   d. In the **`Type`** drop-down list, select **`JKS`** as the keystore type.

   e. In the **`Key file password`** text box, enter the password that you used in step 3(e) of the section "Creating the Certificates and Keystores."

   f. Click **`Get Key File Aliases`** to populate the **`Certificate alias to import`** text box with the server-certificate alias (the short host name). The server-certificate alias is the same name that you specified in step 4 of the section "Creating the Certificates and Keystores."

   g. Click **`OK`**.

   h. Review your entries and save the changes when you are prompted.

3. In the console, import the CA certificate into the node-level trust store, as follows:

   a. Select **SSL certificate and key management ► SSL configurations ► NodeDefaultSSLSettings ► Key stores and certificates ► NodeDefaultTrustStore ► Signer certificates**.

   b. Click **`Add`**.

   c. In the **`Alias`** text box, enter the **`Label`** value that you specified previously in step 7(d) of the section "Creating the Certificates and Keystores."

   d. In the **`File Name`** text box, enter the path to the CA certificate (myCA.crt) that you specified in step 2 of that same section.

   e. Click **`OK`**.

   f. Review your entries and save the changes when you are prompted.

4. Import the signer's certificate (root CA) into the cell-level keystore, as follows:

   a. Select **Security ► SSL certificate and key management ► SSL configurations ► CellDefaultSSLSettings ► Key stores and certificates ► CellDefaultKeyStore ►Signers certificates**.

   b. Click **`Add`**.

   c. In the **`Alias`** text box, enter the **`Label`** value that you specified previously in step 7(d) of the section "Creating the Certificates and Keystores."

d. In the **File Name** text box, enter the path to the CA certificate (myCA.crt) that you specified in step 2 of that same section.

e. Click **OK**.

f. Review your entries and save the changes when you are prompted.

5. Import the signed server certificate from the serverids.jks keystore and the signer's certificate (root CA) into the cell-level keystore, as follows:

   a. Select **Security ► SSL certificate and key management ► SSL configurations ► CellDefaultSSLSettings ► Key stores and certificates ► CellDefaultKeyStore ► Personal certificates ► Import certificates from a key file or key store**.

   b. Click the **Key store file** link.

   c. In the **Key file name** text box, enter the path to the keystore that you created in step 3(c) of the section "Creating the Certificates and Keystores."

   d. In the **Type** list, select **JKS** as the keystore type.

   e. In the **Key file password** text box, enter the password that you used in step 3(e) of the section "Creating the Certificates and Keystores."

   ~~f.~~ Click **Get Key File Aliases** to populate the **Certificate alias to import** text box with the server certificate alias (short host name). This is the same name that you specified in step 4 of the section "Creating the Certificates and Keystores."

   g. Click **OK**.

   h. Review your entries and save the changes when you are prompted.

6. Import the signer's certificate (root CA) into the cell-level trust store, as follows:

   a. Select **Security ► SSL certificate and key management ► SSL configurations ► CellDefaultSSLSettings ► Key stores and certificates ►CellDefaultTrustStore ► Signers certificates**.

   b. Click the **Add** button. .

   c. In the **Alias** text box, enter the **Label** value that you specified previously in step 7(d) of the section "Creating the Certificates and Keystores."

   d. In the **File Name** text box, enter the path to the certificate (myCA.crt) that you specified in step 2 of "Creating the Certificates and Keystores."

   e. Click **OK**.

   f. Review your entries and save the changes when you are prompted.

7. Add the SSL configuration for the SAS server instances.

   a. Select **Security ► SSL certificate and key management SSL ► configurations**.

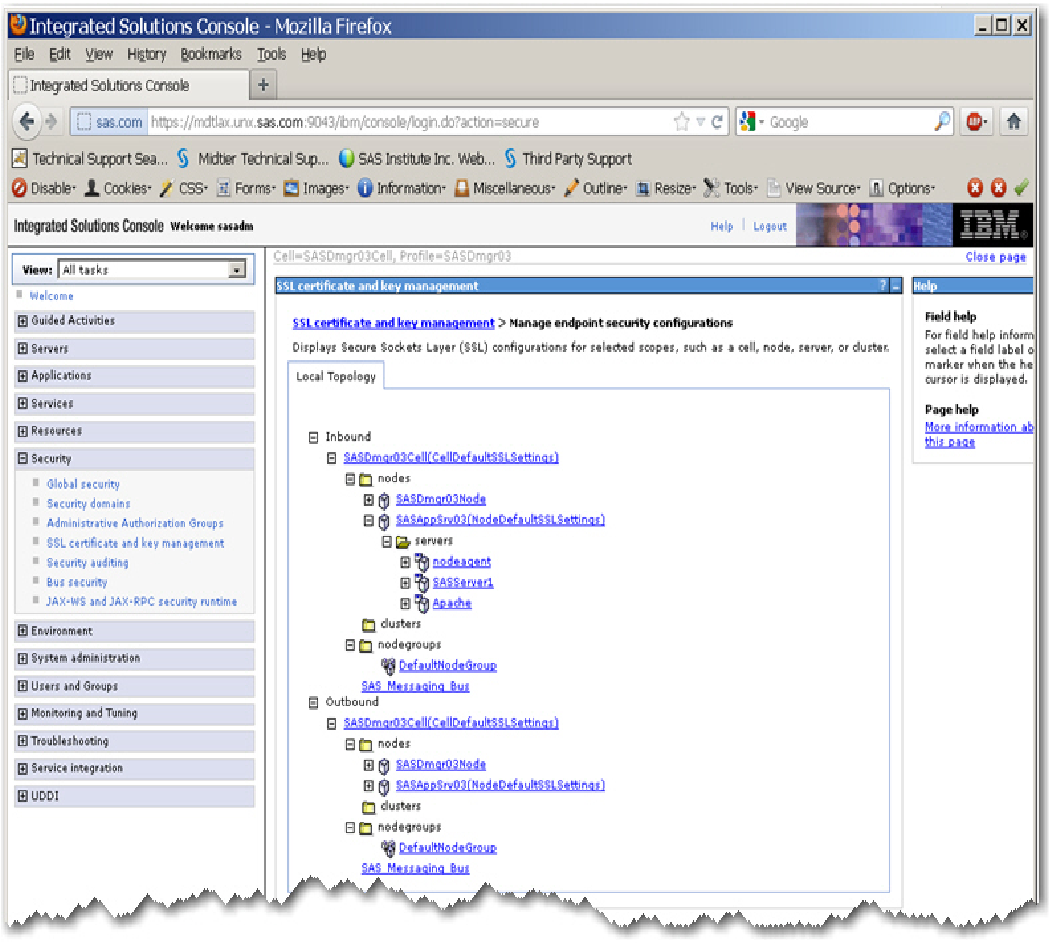
   b. Click **New.**

   c. Enter **SASServer1SSLSettings** in the **Name** text box.

   d. Review your entries and save the changes when you are prompted.

Note: If you have more than one server instance, repeat this process for each one and modify the name accordingly.

8. Update the SSL end-point security configuration settings.
   a. Select **Security ► SSL Certificate and Key Management ► Manage endpoint security configurations**.
   b. Expand the **`Inbound`** node down through the **`servers`** node, as shown in the following display.


Integrated Solutions Console - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Integrated Solutions Console
https://mdtlax.unx.sas.com:9043/ibm/console/login.do?action=secure
Google
Technical Support Sea... Midtier Technical Sup... SAS Institute Inc. Web... Third Party Support
Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options
Integrated Solutions Console Welcome sasadm
Help | Logout
IBM
View: All tasks
Welcome
Guided Activities
Servers
Applications
Services
Resources
Security
Global security
Security domains
Administrative Authorization Groups
SSL certificate and key management
Security auditing
Bus security
JAX-WS and JAX-RPC security runtime
Environment
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Service integration
UDDI
Cell=SASDmgr03Cell, Profile=SASDmgr03
Close page
SSL certificate and key management
Help
Field help
For field help inform select a field label o marker when the he cursor is displayed.
Page help
More information ab this page

SSL certificate and key management > Manage endpoint security configurations

Displays Secure Sockets Layer (SSL) configurations for selected scopes, such as a cell, node, server, or cluster.

Local Topology

```
Inbound
  SASDmgr03Cell(CellDefaultSSLSettings)
    nodes
      SASDmgr03Node
      SASAppSrv03(NodeDefaultSSLSettings)
        servers
          nodeagent
          SASServer1
          Apache
    clusters
    nodegroups
      DefaultNodeGroup
    SAS Messaging Bus
Outbound
  SASDmgr03Cell(CellDefaultSSLSettings)
    nodes
      SASDmgr03Node
      SASAppSrv03(NodeDefaultSSLSettings)
    clusters
    nodegroups
      DefaultNodeGroup
    SAS Messaging Bus
```

   c. Click **`SASServer1`**.
   d. Change the SSL configuration property under **`Specific SSL Configuration for this endpoint`** to **`SASServer1SSLSettings`**.
   e. Click the **`Update certificate alias`** list. Make sure that new server identity certificate shows in the **`Certificate alias for keystore`** list.
   f. Select the **`Override inherited values`** check box because the keystore and the trust store are defined at the node level.
   g. Review or Save the changes when you are prompted.

Note: If you have more than one SAS server instance, repeat this process for each one and modify the name accordingly.

9. If you previously disabled WebSphere administrative security, re-enable it now. Verify that the **`Enable application security`** check box is checked as well if it was previously enabled before you started this configuration.
10. Select **System Administration ► Nodes** and synchronize the nodes, if it is necessary. Depending on how your WebSphere console is configured, this synchronization might have been done automatically when you saved changes during the previous steps.

Configuring SAS Remote Services for One-Way SSL

You need to specify the path to the trusted CA keystore and its associated password for SAS Remote Services. You specify the location of the trusted CA keystore and its password in the following Java virtual machine (JVM) parameters, which you must add to the wrapper.conf file:

- If SAS Remote Services is installed as a service, add the following parameters to wrapper.conf, which is located in ***`SAS-configuration-directory`*****`\ Lev1\Web\Applications\RemoteServices`**.

```
wrapper.java.additional.13=-Djavax.net.ssl.trustStore="path-to-keystore\
                                                        trustedca.jks"

wrapper.java.additional.14=-Djavax.net.ssl.trustStorePassword=password
```

  Note: The number specified as *nn* in **`wJavaAgs_nn=`** are sequential numbers, so the numbers you see might differ from the numbers (13 and 14) that are shown in the example above.

- If SAS Remote Services is started from a batch file, add the following parameters to both the **`start2`** and **`start3`** sections in the RemoteServices.bat file or the RemoteServices.sh file:

```
-Djavax.net.ssl.trustStore="path-to-keystore \trustedca.jks"

-Djavax.net.ssl.trustStorePassword=password
```

Updating the SAS Metadata

Configuring the Connection Metadata for SAS Web Applications

You will communicate with your SAS web applications via HTTPS. However, first you must use the Configuration Manager plug-in in SAS Management Console to change the connection information for the SAS Logon Manager and the other SAS web applications (for example, SAS® Information Delivery Portal) to which you want to connect via HTTPS You need to change the connection information because in SAS 9.3 the SAS Logon Manager does not direct you to applications by using information from a URL that you have entered in the browser (other than the application name). Instead, the logon manager uses information that is stored in the SAS® Metadata Repository. Each application is represented by a metadata object that has properties such as a communication protocol, a host name, a port number, and a service name (context root).

To configure the connection information for SAS web applications:

1. Open SAS Management Console.
2. On the **`Plug-ins`** tab in the left pane, select **Application Management ► Configuration Manager ► SAS Application Infrastructure**.
3. Right-click **`Logon Manager 9.3`** and select **`Properties`** from the pop-up menu.
4. In the Logon Manager 9.3 Properties dialog box, click the **`Connection`** tab.
5. On the **`Connection`** tab, set **`Communication Protocol`** to **`HTTPS`** and **`Port Number`** to **`9443`**. Then click **`OK`**.
6. Repeat steps 2 through 5 for the SAS web applications that you want to log on to via HTTPS.

SAS Themes and SAS Help both consist of static content. You might want to configure these applications for HTTPS, depending on your performance and security requirements. If you configure these applications for HTTP only, you will receive mixed-content warnings from the browser, which might not be acceptable in your environment.

To configure SAS Themes and SAS Help Viewer for SSL:

1. On the **`Plug-ins`** tab in SAS Management Console, select **Application Management ► Configuration Manager**.
2. In the right pane, right-click the entry that you want (either **`SASTheme_default`** or **`Help Viewer for Midtier App 9.3`**) and select **`Properties`** from the pop-up menu.

   Note: For this example, **`SASTheme_default`** is selected.
3. In the Properties dialog box, click the **`Connection`** tab.
4. On the **`Connection`** tab, set **`Communication Protocol`** to **`HTTPS`** and **`Port Number`** to **`9443`**. Then click **`OK`**.

Configuring the Connection Metadata for the SAS® Content Server

To configure the connection metadata for the SAS Content Server:

1. In SAS Management Console, select **Plug-ins ► Environment Manager ► Server Manager ► SAS Content Server**. Expand **`SAS Content Server`** so that **`Connection:SAS Content Server`** appears in the right pane of SAS Management Console.
2. Right-click **`Connection: SAS Content Server`** entry and select **`Properties`**.
3. On the **`Options`** tab in the SAS Content Server Properties dialog box, modify the **`Application protocol`** field to display **`https`**.
4. For **`Port number`**, enter the correct port number. The default port number for this example is **`8443`**.
5. Click **`OK`** to save your settings and exit the Connection: SAS Content Server Properties dialog box.
6. In SAS Management Console, select **Folders ► SAS Folders**.
7. Right-click **`SAS Folders`** and select **`Properties`**.

8. In the SAS Folders Properties dialog box, click the **`Content Mapping`** tab.
9. In the **`Server`** list, select **`SAS Content Server`**.
10. Click **`OK`** to exit the dialog box.

Note: These changes go into effect after you restart the WebSphere server instance.

SAS Management Console enables you to validate your connection to the SAS Content Server. However, if SAS Management Console cannot locate the certificate, you might see the following exception message when you right-click **`SAS Content Server`** and select **`Validate`** from the menu:

```
(sun.security.validator.ValidatorException: PKIX path building failed:

sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid
certification path to requested target)
```

If the content server cannot locate the trusted certificate, you need to add the following two JVM parameters to the smc.ini file. The smc.ini file is located in the installation directory for SAS Management Console. The JVM parameters below are the same parameters that you added previously in the section “Configuring SAS Remote Services for One-Way SSL.”

```
JavaArgs_15=-Djavax.net.ssl.trustStore="path-to-keystore\trustedca.jks"

JavaArgs_16=-Djavax.net.ssl.trustStorePassword=password
```

Note: The number specified as *nn* in **`wJavaAgs_`*`nn`*`=`** are sequential numbers, so the numbers you see might differ from the numbers (15 and 16) that are shown in the example above.

After you add these parameters, the SAS Content Server connection should validate as expected.

Configuring the WebDAV URL for the SAS® Foundation Services Repository

Now, you need to configure the WebDAV URL for HTTPS in SAS Management Console for the following application services:

- **`Remote Services`**
- **`SASBIPortlet4.3 Local Services`**
- **`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:

1. In SAS Management Console, select **Plug-ins ► Environment Management ► Foundation Services Manager ►*application-service-name* ►Core ► Information Service**.
2. Right-click **`Information Service`** and select **`Properties`**.
3. In the Information Service Properties dialog box that opens, click the **`Service Configuration`** tab.
4. On the **`Service Configuration,`** click **`Configuration`** to open the Information Service Configuration dialog box.
5. In the dialog box, click the **`Repositories`** tab.
6. Select **`WebDAV`** and then click **`Edit`**.
7. Change the **`Port`** value to the SSL port that you have configured for your application server (for example, **`9443)`**. The host name should already be set to the web application server that is hosting the SAS Content Server; that is, the host on which the sas.wip.scs9.3.ear file is deployed.
8. Select the **`Secure`** check box.
9. Click **`OK`** to close the Information Service Configuration dialog box.
10. Click **`OK`** to close the Information Service Properties dialog box.

Verifying the SSL Connection

If you are using this document to configure one-way SSL with host authentication, that process is complete after you verify your connection. To verify your SSL connection, stop and then restart SAS Remote Services and the WebSphere SAS domain servers on the server where you made the updates for SSL (for example, on SASServer1). At this point, you should be able to log on to your SAS application by entering **`https://`*`fully-qualified-hostname`*`:9443/`*`SAS-application-name`*** in the browser. (An example of the value for ***`SAS-application-name`*** is **`SASPortal`**). The next section explains how to configure your environment with two-way SSL and client-certificate authentication.

Configuring Two-Way Secure Socket Layer

Deploying SAS web applications to use client-certificate web server authentication requires configuration of the web server to support two-way SSL. With *one-way SSL* only the identity of the server is verified; while in *two-way SSL*, the identities of both the server and the client are verified.

The following sections cover these topics related to the steps for configuring two-way SSL:

- creating and importing personal certificates into your browser
- configuring WebSphere for two-way SSL
- configuring SAS Remote Services for two-way SSL
- configuring client-certificate authentication for WebSphere

Creating Personal Client Certificates

You must create personal certificates for each client person or identity that needs to be verified.

To create these personal certificates:

1. Create a personal private key (a PEM file) for the client identity.

   ```
   openssl genrsa -out sasdemo.pem -des3  1095
   ```

2. Generate the Personal Private Key x.509 request file (a CSR file) with DN:

   ```
   openssl req -new -days 1095 -key sasdemo.pem –out sasdemo.csr -subj

   /CN=sasdemo/UID=sasdemo/OU=organization-unit/O=organization
             /C=US/emailAddress=e-mail@domain.com/
   ```

3. Sign the certificate with the CA certificate, as follows:

   ```
   openssl x509 -req -days 1095 -in sasdemo.csr -CA myCA.crt
                 -CAkey myCAkey.pem -CAcreateserial -out sasdemo.crt
   ```

4. Combine the public and private key into PCKS12 format.

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

   You will be prompted to provide an export password.

5. Import the client certificate into the browser in the **`Personal`** area for Internet Explorer and into the **`Your Certificates`** area for Firefox as described in the next section.

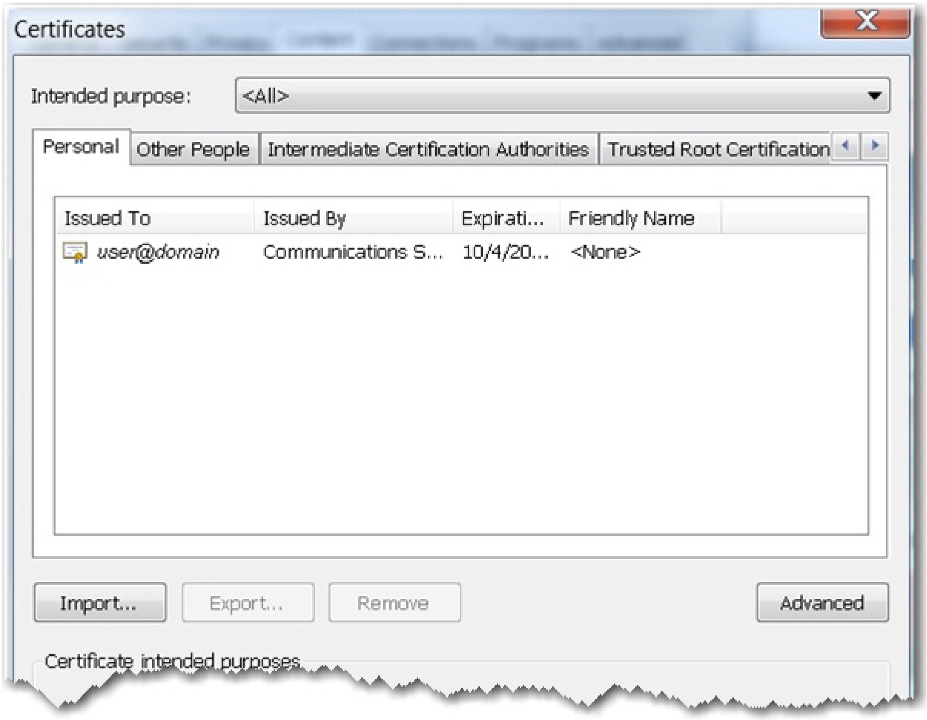
You now have the certificates that you need for two-way SSL.

Importing the Personal Certificate into Your Browser

During one-way SSL configuration, you imported the CA certificate into the trusted-certificate area of your browser. For two-way SSL, you also need to import the certificate that you just created to identify for client verification during the SSL message exchange (handshake). First, copy the certificate to the machine where you access the browser. Then follow the instructions below for your particular browser.

To import this certificate into Internet Explorer:

1. Select **Tools** ► **Internet options** ► **Content** ► **Certificates** ► **Personal**.

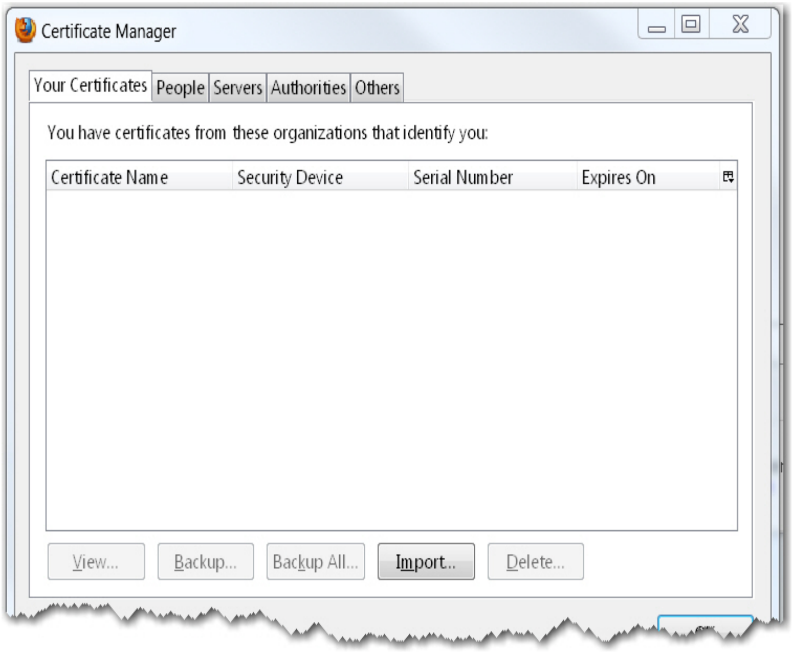

Certificates
Intended purpose:
<All>
Personal
Other People
Intermediate Certification Authorities
Trusted Root Certification
Import...
Export...
Remove
Advanced
Certificate intended purposes

Issued To	Issued By	Expirati...	Friendly Name
user@domain	Communications S...	10/4/20...	<None>

2. Click the **`Import`** button to start the Certificate Import Wizard. Follow the wizard's instructions. You need to supply the path to the certificate file and the password. For any other values, accept the defaults.

To import this certificate into Mozilla Firefox:

1. Select **Tools** ► **Options** ► **View Certificates** ► **Your Certificates**.



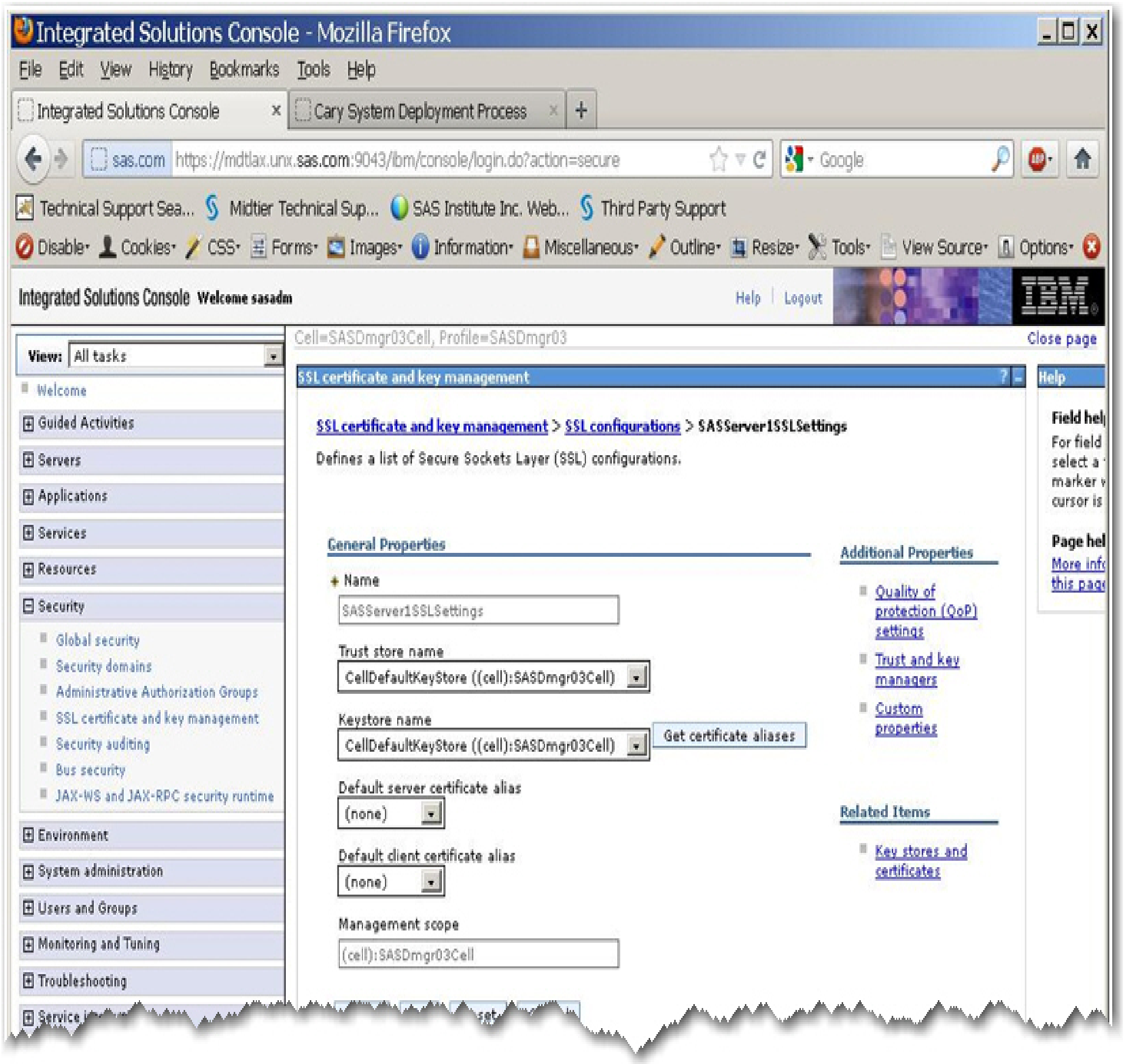
2. Click the `Import` button to start the Certificate Import Wizard. Follow the wizard's instructions. You need to supply the path to the certificate file and the password. For any other values, accept the defaults.

You now have the certificates that you need for two-way SSL.

Configuring WebSphere for Two-Way SSL

To configure WebSphere for two-way SSL:

1. In the WebSphere administration console, select **Security ► SSL certificate and key management ► SSL configurations ► SASServer1SSLSettings**.


Integrated Solutions Console - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Integrated Solutions Console
Cary System Deployment Process
https://mdtlax.unx.sas.com:9043/ibm/console/login.do?action=secure
Integrated Solutions Console Welcome sasadm
Help Logout
View: All tasks
Welcome
Guided Activities
Servers
Applications
Services
Resources
Security
Global security
Security domains
Administrative Authorization Groups
SSL certificate and key management
Security auditing
Bus security
JAX-WS and JAX-RPC security runtime
Environment
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Close page
Get certificate aliases
Additional Properties
Quality of protection (QoP) settings
Trust and key managers
Custom properties
Related Items
Key stores and certificates

Cell=SASDmgr03Cell, Profile=SASDmgr03

SSL certificate and key management

SSL certificate and key management > SSL configurations > SASServer1SSLSettings

Defines a list of Secure Sockets Layer (SSL) configurations.

General Properties

* Name: SASServer1SSLSettings

Trust store name: CellDefaultKeyStore ((cell):SASDmgr03Cell)

Keystore name: CellDefaultKeyStore ((cell):SASDmgr03Cell)

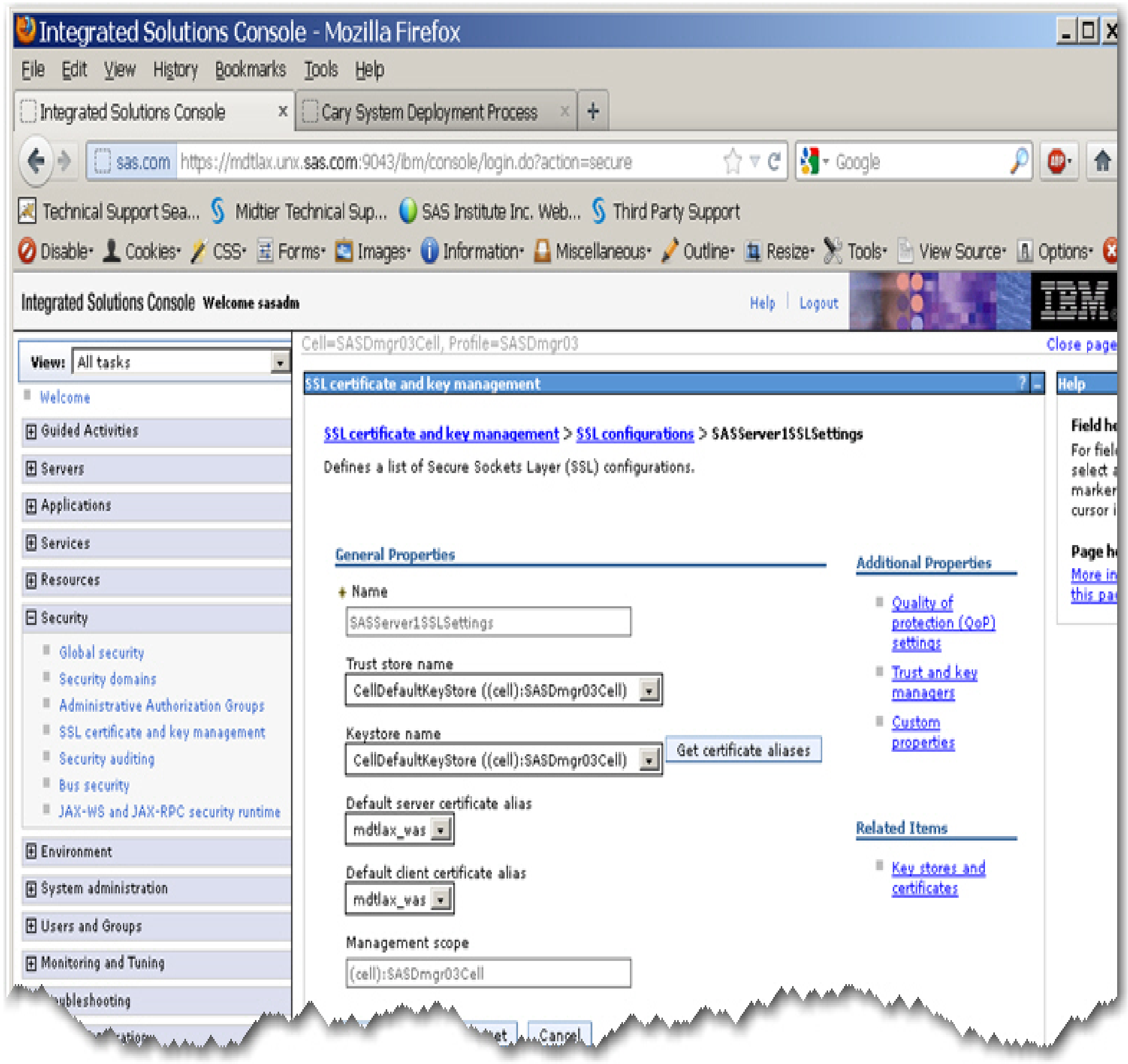
Default server certificate alias: (none)

Default client certificate alias: (none)

Management scope: (cell):SASDmgr03Cell

2. Specify the trust-store name and the keystore name from their associated drop-down lists. You can use the cell-level default keystore and trust-re values that are already populated in the list. Alternatively, you can add the server IDs and the trusted CA JKS stores (created previously) and then select them. For this example, however, the cell-level default values are sufficient because they contain identical root and server certificates.

3. Click the **`Get certificate aliases`** button, as shown in the following display. The default server and client-certificate alias list should be prepopulated with the host-name certificate alias.


Integrated Solutions Console - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Integrated Solutions Console
Cary System Deployment Process
sas.com https://mdtlax.unx.sas.com:9043/ibm/console/login.do?action=secure
Google
Technical Support Sea... Midtier Technical Sup... SAS Institute Inc. Web... Third Party Support
Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options
Integrated Solutions Console Welcome sasadm
Help | Logout
IBM
View: All tasks
Welcome
Guided Activities
Servers
Applications
Services
Resources
Security
Global security
Security domains
Administrative Authorization Groups
SSL certificate and key management
Security auditing
Bus security
JAX-WS and JAX-RPC security runtime
Environment
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Close page
Help
Field help
Page help
More information about this page

Cell=SASDmgr03Cell, Profile=SASDmgr03

SSL certificate and key management

SSL certificate and key management > SSL configurations > SASServer1SSLSettings

Defines a list of Secure Sockets Layer (SSL) configurations.

General Properties

* Name: SASServer1SSLSettings

Trust store name: CellDefaultKeyStore ((cell):SASDmgr03Cell)

Keystore name: CellDefaultKeyStore ((cell):SASDmgr03Cell) [Get certificate aliases]

Default server certificate alias: mdtlax_vas

Default client certificate alias: mdtlax_vas

Management scope: (cell):SASDmgr03Cell

Additional Properties

- Quality of protection (QoP) settings
- Trust and key managers
- Custom properties

Related Items

- Key stores and certificates

4. In the administration console, select **Security ►SSL certificate and key management ► SSL configuration ► SASServer1SSLSettings.** Then click the **`Quality of protection (QoP)`** settings link on the right.

5. Select **`Required`** from the **`Client authentication`** list.

6. Review your entries and save the changes when you are prompted.

Configuring SAS Remote Services for Two-Way SSL

You need to specify the path to the server-identity keystore and its associated password in the wrapper.conf file for SAS Remotes Services. You specify the location and name of the server-identity keystore and its password in the JVM parameters that you add to wrapper.conf, as follows:

- If SAS Remote Services is installed as a service, add the following parameters to the wrapper.conf file, which is located in ***SAS-configuration-directory*\ Lev1\Web\Applications\RemoteServices**:

```
wrapper.java.additional.15=-Djavax.net.ssl.keyStore="path-to-keystore\
                                                       serverids.jks"
wrapper.java.additional.16=-Djavax.net.ssl.keyStorePassword=password
```

- If you start SAS Remote Services from a batch file, add the following parameters both to the **start2** and **start3** sections in the RemoteServices.bat or RemoteServices.sh files:

```
-Djavax.net.ssl.keyStore="path-to-keystore\serverids.jks"
-Djavax.net.ssl.keyStorePassword=password
```

In either case, you can add these parameters directly after the **trustStore** and **trustStorePassword** parameters that were added for the one-way SSL configuration.

Verifying the SSL Connection

If you are using this document to configure two-way SSL with host authentication, that process is complete after you verify your connection. To verify your SSL connection, stop and then restart SAS Remote Services and the WebSphere SAS domain servers on the server where you made the updates for SSL (for example, on SASServer1). At this point, you should be able to log on to your SAS application by entering **https:*//fully-qualified-hostname:*9443*/SAS-application-name*** in the browser. (An example of the value for ***SAS-application-name*** is **SASPortal**).

Configuring WebSphere for Web Authentication

The default security mechanism for SAS web applications is to authenticate against the authentication provider of the SAS® Metadata Server. As an alternative authentication mechanism, you can use web authentication. With this method, you configure WebSphere to authenticate against a user registry such as a Lightweight Directory Access Protocol (LDAP) server, and you configure SAS web applications to trust the authentication that WebSphere performs. For client-certificate authentication, the environment must first be configured for web authentication. To configure your system for web authentication, follow the steps in *Configuring IBM WebSphere Application Server 7.0 for Web Authentication with SAS® 9.3 Web Applications*. (**support.sas.com/resources/thirdpartysupport/v93/appservers/ConfiguringWAS7WebAuth.pdf**)

Configuring WebSphere for Client-Certificate Authentication

To configure client-certificate authentication for WebSphere:

1. In the WebSphere administration console, select **Global security ► Standalone LDAP registry.**
2. Click **Configure**.
3. Navigate to **Advanced Lightweight Directory Access Protocol (LDAP) user registry settings.**

4. Set the certificate-map mode to **`CERTIFICATE_FILTER`**, where the filter value is **`uid=${SubjectCN}`**. This setting enables the CN to be matched to the UID field for user identity in the LDAP directory.
5. Review your entries and save the changes when you are prompted.

Configure SAS Logon Manager for Client-Certificate Authentication

To configure the SAS Logon Manager:

1. Change the SAS Logon Manager application's web.xml file so that it contains the **`CLIENT-CERT`** authentication method. The web.xml file resides in the sas.svcs.logon.war file that is packaged with the sas.wip.apps9.3.ear file. Details about the location of these files and how to access them are available in *Configuring IBM WebSphere Application Server 7.0 for Web Authentication with SAS® 9.3 Web Applications*. (**`support.sas.com/resources/thirdpartysupport/v93/appservers/ConfiguringWAS7WebAuth.pdf`**)

   To modify web.xml, add the following method prior to the closing **`</webapp>`** tag:

```
<security-constraint>
    <web-resource-collection>
       <web-resource-name>All resources</web-resource-name>
       <url-pattern>/*</url-pattern>
       <http-method>GET</http-method>
       <http-method>POST</http-method>
    </web-resource-collection>
    <auth-constraint>
       <role-name> SASWebUser </role-name>
    </auth-constraint>
</security-constraint>

<login-config>
   <auth-method>CLIENT-CERT</auth-method>
   <realm-name>default</realm-name>
</login-config>

<security-role>
   <role-name>SASWebUser</role-name>
</security-role>
```

2. Rebuild the sas.svcs.logon.war and sas.wip.apps9.3.ear files, and redeploy the EAR file, as explained in *Configuring IBM WebSphere Application Server 7.0 for Web Authentication with SAS® 9.3 Web Applications*. (**`support.sas.com/resources/thirdpartysupport/v93/appservers/ConfiguringWAS7WebAuth.pdf`**)
3. Test the client-certificate authentication, as follows:
   a. Restart the WebSphere deployment-manager process, the node-agent process, SAS Remote Services, and the WebSphere SAS server instance.
   b. Verify that you can access the SAS applications through the URL without having to provide logon credentials. Your browser might prompt you for the specific certificate that is to be used for authentication if more than one personal certificate is installed. For testing purposes, select the **`sasdemo`** personal certificate that you created previously in this document.

Resources

IBM Corporation. (2009). “Creating keystores” in *WebSphere Application Server v7.0 Security Guide.* Armonk, NY. Available at **www.redbooks.ibm.com/redbooks/pdfs/sg247660.pdf**.

Open SSL Software Foundation (2010). “OpenSSL Cryptography and SSL/TLS Toolkit.” Available at **www.openssl.org/**.

SAS Institute Inc. (2011). *Configuring IBM WebSphere Application Server 7 for Web Authentication with SAS® 9.3 Web Applications*. Cary, NC. Available at **support.sas.com/resources/thirdpartysupport/v93/appservers/ConfiguringWAS7WebAuth.pdf**.

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.

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.