Configuring HP Service Manager to Use the SSL-based Trusted Sign-On and LW-SSO

Table of Contents

Introduction

The Service Manager Server uses a proprietary Single Sign-On (SSO) protocol that is based on mutually-authenticated Secure Sockets Layer (SSL).

When an HP application (for example, HP BSM Operations Manager i) needs to access the Service Manager Web tier using LW-SSO, Service Manager must be configured to use SSL-based Trusted Sign-On and to support LW-SSO.

When an HP application (for example, HP Release Control) needs to access Service Manager Web Services using LW-SSO, an extra step is required to install and configure the Symphony Adapter (for Web services), which cannot work without SSL.

The following table lists the configuration procedures required in Service Manager for HP applications that need to access the Service Manager Web tier or Web Services using LW-SSO.

Example Application	**Need to access Service Manager through**	**Configurations required in Service Manager**
HP BSM Operations Manager i	Web tier	• Configure Service Manager to use SSL-based Trusted Sign-On • Configure Service Manager for LW-SSO support
HP Release Control	Web Services	• Configure Service Manager to use SSL-based Trusted Sign-On • Install and configure Symphony Adapter • Configure Service Manager for LW-SSO support

This document describes the detailed steps for these configuration procedures in Service Manager 9.20.

Configuring Service Manager to Use SSL-based Trusted Sign-On

The following procedures are provided as examples, assuming that you have OpenSSL 1.0.0 installed in your system.

NOTES:

- The following procedures will prompt you to enter several passwords multiple times. Using the same password over and over is not best practice in production, however if you are performing the procedures for test purposes, you are recommended to enter the same password at each prompt to avoid any confusion about which password you are being asked for. Also be aware that nothing displays on the screen when you are entering certain pass phrases.
- Whenever asked to confirm whether to trust the current certificate, type **y** and press ENTER (the default response is no). If you just press ENTER, the certificate will not be trusted and you will have to start over.

Task 1: Create a root CA

Open a command window in your system, and run the command line for each step:

1. Generate a CA key pair:

   ```
   openssl genrsa -des3 -out ca.key 2048
   ```

   When prompted, enter a pass phrase for ca.key. You will be asked to enter this pass phrase later many times again.

2. Export the public key as the self-signed root CA certificate:

   ```
   openssl req -new -key ca.key -x509 -days 1095 -out ca.cer
   ```

   a. When prompted, enter the pass phrase you selected for ca.key.

   b. Enter other required information. When asked for a Common Name, enter whatever you like, for example, your name.

3. Import the root CA certificate into a keystore file:

   ```
   keytool -import -alias cacer -file ca.cer -keystore ca.keystore
   ```

   a. Enter a password for ca.keystore. Write down this password (<root CA keystore pass phrase>), which will be added to the sm.ini file later.

   b. When asked to confirm whether to trust the certificate, type: **y**.

   The root CA certificate is added to the ca.keystore.

Task 2: Set up the Service Manager server

Subtask 1: Create a keystore for the Service Manager server

1. Generate a private/public key pair:

   ```
   keytool -genkey -alias smserver -keystore smserver.keystore
   ```

   a. Enter a password for the server keystore (<Service Manager server keystore password>). Write down this password, which will be added to the sm.ini file later.

   b. When prompted to enter your first and last name, enter the fully-qualified domain name (computer.domain.com) of the Service Manager server host.

   c. Enter other required information.

   d. Enter a key password for <smserver>.

2. Generate a request file:

   ```
   keytool -certreq -alias smserver -keystore smserver.keystore -file
   smserver_cer.crs
   ```

   Enter the password you selected for the server keystore in step 1 of this subtask.

3. Self-sign the request:

   ```
   openssl x509 -req -days 1095 -in smserver_cer.crs -CA ca.cer -CAkey
   ca.key -CAcreateserial -out smserver.cer
   ```

   If everything goes well, a message “Signature ok” displays. When prompted, enter the pass phrase for ca.key.

4. Import the root CA certificate into the server keystore:

```
keytool -import -alias cacer -file ca.cer -keystore smserver.keystore
```

a. Enter the password you selected for the server keystore in step 1 of this subtask.

b. Type **y** to confirm that you want to trust the certificate.

The root CA certificate is added to the server keystore.

5. Import the signed certificate into the server keystore:

```
keytool -import -alias smserver -file smserver.cer -keystore
smserver.keystore
```

Enter the password you selected for the server keystore in step 1 of this subtask.

The signed certificate is installed in the server keystore.

Subtask 2: Create a trusted client keystore

This task is to create a client keystore for any application that needs to connect to the server host. If your web tier and Windows clients are deployed on the same host, they can share one client keystore; if not, you need to create a client keystore for each client.

1. Generate a private/public key pair:

```
keytool -genkey -alias client -keystore client.keystore
```

a. Enter a password for the client keystore (client.keystore).

b. When prompted to enter your first and last name, always enter the fully-qualified domain name (computer.domain.com) of the web tier (or Windows client) host.

c. Enter a key password for <client>.

2. Generate a request file:

```
keytool -certreq -alias client -keystore client.keystore -file
client_cer.crs
```

Enter the client keystore password.

3. Self-sign the request:

```
openssl x509 -req -days 1095 -in client_cer.crs -CA ca.cer -CAkey
ca.key -CAcreateserial -out client.cer
```

If everything goes well, a message “Signature ok” displays.

When prompted, enter the pass phrase for ca.key.

4. Import the root CA certificate into the client keystore:

```
keytool -import -alias cacer -file ca.cer -keystore client.keystore
```

Enter the client keystore password, and type **y** to trust the certificate.

5. Import the signed certificate into the client keystore:

```
keytool -import -alias client -file client.cer -keystore
client.keystore
```

When prompted, enter the client keystore password.

A message displays: “Certificate reply was installed in keystore”.

Subtask 3: Generate a trust-list keystore for the Service Manager server

This task applies for both the Windows and web clients. If the web tier and Windows clients are deployed on the same host, they can share the same client keystore and you only need to perform this task once; otherwise, you need to repeat this step for each client certificate.

1. Import the client certificate you created in subtask 2 into a jks file:

   ```
   keytool -import -alias client1 -file client.cer -keystore
   trusted.keystore
   ```

2. When prompted, enter a password for the trusted keystore (<trusted client keystore password>. **NOTE**: Write down this password, which will be added to the sm.ini file later.

3. When asked, type **y** to confirm that you want to trust the certificate.

   A message displays: "Certificate was added to keystore".

Subtask 4: Modify the Service Manager configuration

1. Copy the generated files ca.keystore, smserver.keystore, and trusted.keystore into the <Service Manager installation path>\Server\RUN\ folder.

2. In the sm.ini file, set the sslConnector parameter to 1 if it is 0.

3. Add the following entries to the sm.ini file:

   ```
   keystoreFile:smserver.keystore
   keystorePass:<Service Manager server keystore password>
   ssl:1
   ssl_reqClientAuth:2
   ssl_trustedClientsJKS:trusted.keystore
   ssl_trustedClientsPwd:<trusted client keystore password>
   trustedsignon:1
   truststoreFile:ca.keystore
   truststorePass: <root CA keystore pass phrase>
   ```

4. Restart the Service Manager server.

Task 3: Set up the Service Manager web tier

This task is to create a client certificate for any application that needs to connect to the Service Manager server using the Service Manager trusted sign-on protocol.

The following steps assume that your Service Manager web tier is deployed on Tomcat.

1. Copy the CA keystore file "ca.keystore" to the webapps\<Service Manager web tier>\WEB-INF folder of the Tomcat installation.

2. Copy the client keystore file "client.keystore" to the webapps\<Service Manager web tier>\WEB-INF folder of the Tomcat installation.

3. Edit <Tomcat>\webapps\<Service Manager web tier>\WEB-INF\web.xml.

   a Configure the Service Manager web tier to use the client certificate you have just created.

   You have to specify three parameters as shown below:

   - the location of the CA keystore file created previously

- the location of the client keystore just created
- the password for the client keystore

```
<!-- Specify the CA certificate store to use in encrypted
communication -->
<init-param>
<!-- If this value is empty, the JDK's default
jre/lib/security/cacerts file is used -->
<!-- If this is a relative path, it will be relative to the web
application's deploy directory
but still needs a leading slash -->
<param-name>cacerts</param-name>
<param-value>/WEB-INF/ca.keystore</param-value>
</init-param>


<!-- Specify the client's private keystore to use in encrypted
communication. This is necessary
for client authentication when using single sign-on, but not for a
standard SSL connection. -->
<!-- If this is a relative path, it will be relative to the web
application's deploy directory
but still needs a leading slash -->
<init-param>
<param-name>keystore</param-name>
<param-value>/WEB-INF/client.keystore</param-value>
</init-param>


<!-- Specify the password for the client's private keystore -->
<init-param>
<param-name>keystorePassword</param-name>
<param-value><client keystore password></param-value>
</init-param>
```

b In the web.xml file, make sure the server FQDN name is placed instead of 'localhost'.

```
<!-- Specify the HP Service Manager server host and port location
-->
    <init-param>
      <param-name>serverHost</param-name>
      <param-value><Service Manager server host name
(FQDN)></param-value>
    </init-param>
```

c Make sure ssl is set to true.

```
<!-- Control the encryption of network communication between the
application server
         and the HP Service Manager server -->
    <init-param>
      <param-name>ssl</param-name>
      <param-value>true</param-value>
    </init-param>
```

d Enable drill from BAC to Service Manager.

Set the following in case an error message (cracking attempt) comes up while drilling to Service Manager from the EMS monitor.

```
<init-param>
      <param-name>querySecurity</param-name>
      <param-value>false</param-value>
</init-param>
```

e Restart Tomcat.

When the configuration is complete and the Tomcat container has been restarted, the Service Manager web tier is enabled to use trusted sign-on when communicating with the Service Manager server.

NOTE: If you start a web browser from your desktop and start the web client, it will still display the log-in panel.

Task 4: Set up the Service Manger Windows client to enable SSL connection

After SSL-based Trusted Sign-On is enabled for the Service Manager server, you can enable an SSL connection for the Windows client so that Windows client users can connect to the server.

Note: Steps 1 through 4 below assume that the Windows client and the web tier are deployed on different hosts. If they are deployed on the same host, you can skip these steps and copy the CA keystore file and the web client keystore file to the Windows client host instead.

1. Copy the CA keystore file "ca.keystore" to a local directory on the host where the Windows client is installed. For example, C:\ssl.
2. Open a command window on the host and change to this directory.
3. Repeat subtask 2 to create a client keystore for the Windows client.
4. On the Service Manager server host, repeat subtask 3 to import the Windows client keystore to the trusted.keystore file in the <SM installation path>\Server\RUN\ folder.
5. Restart the Service Manager server.
6. Start the Service Manager Windows client.
7. Set the Windows client security preferences.
   a From the menu bar, select Window > Preferences… to open the Preferences dialog.
   b Expand the **HP Service Manager** node in the menu tree, and select **Security** to open the client security dialog.
   c In the **CA certificates file** field, browse to the ca.keystore file. For example: C:\ssl\ca.keystore.
   d In the **Client keystore file** field, browse to the client.keystore file. For example, C:\ssl\client.keystore.
   e In the **Client keystore password** field, enter the password you specified for the Windows client's client.keystore file.

8. Update the server connections.
   a Go to **File** > **Connect** > **Connections**.
   b In the Connections tab, make sure that the **Server host name** field value is the FQDN of the Service Manager server host.
   c In the **Advanced** tab, select **Use SSL Encryption**.
   d Click **Apply**.
9. Restart the Windows client for the new security configuration to take effect.
10. Log on to the Windows client without using Trusted Sign-On.

    If you can log on successfully, the SSL configuration for the Service Manager server is successful.

Installing and configuring Symphony Adapter

This section describes installation and configuration of the latest version of Symphony Adapter for Service Manager.

Symphony Adapter resides between another HP application and Service Manager to convert LW-SSO to Service Manager Trusted Sign-on SSO protocol.

- If you are setting up your environment for an application to access the Service Manager Web Services using LW-SSO, you must install and configure Symphony Adapter.
- **IMPORTANT**: If you are setting up your environment for an application to access the Service Manager Web tier using LW-SSO, you do NOT need to install and configure Symphony Adapter. Symphony Adapter is bypassed in this case.

Installation

The adapter is provided as a .war file, intended to be deployed in a suitable web container. It is recommended that the SymphonyAdapter.war file be deployed in the same Tomcat container in which the Service Manager Web tier has been deployed. This allows the web tier and Symphony Adapter to share the same client certificate.

NOTE: Before proceeding to the following steps, make sure that the Service Manager Windows and Web clients are still working properly following the configuration steps described above.

1. Copy and unzip the latest adapter file SymphonyAdapter.zip from the Service Manager DVD to a local drive on your Service Manager web tier host.
2. Deploy the SymphonyAdapter file in the Tomcat container, by copying and pasting the SymphonyAdapter.war file into the Tomcat webapps directory.

Configuration

1. Copy the CA keystore file “ca.keystore” and web client keystore file client.keystore you created into the <symphonyadapter>\WEB-INF\classes folder of the SymphonyAdapter webapp.
2. Edit the SymphonyAdapter.properties file to correct the settings for your installation as described in the following table.

Setting	Description
servicecenter.ws.targetLocation URL	This property setting can be left alone if the Service Manager server is running on the same host. Otherwise, edit the hostname. Leave the port and path alone. For example: servicecenter.ws.targetLocationURL= https://<Service Manager server host name (FQDN)>:13443/sc62server/ws
servicecenter.webtier.URL	Update this property to make the hostname and port correct for the current Tomcat container. **NOTE**: Do NOT Specify LOCALHOST. You must provide the full host name. For example: http://<Service Manager web tier host name (FQDN)>:8080/sm920/index.do
clientcerts.keystore	Update this parameter to point to the Symphony Adapter client keystore you created. **NOTE**: For this parameter, you must use a full path name starting from C: and using double slashes, for example: C:\\Program Files\\Apache Software Foundation\\Tomcat 5.5\\webapps\\SymphonyAdapter\\WEB-INF\\classes\\client.keystore **NOTE**: Do not put any newline characters in the path. The example above is just wrapped in this document due to the length of the line.
clientcerts.keystore.password	Specify the correct pass-phrase for the client keystore specified above. For example: !qaz2wsx3edc
truststore	Specify the full path to the CA keystore file you created. For example: C:\\Program Files\\Apache Software Foundation\\Tomcat 5.5\\webapps\\SymphonyAdapter\\WEB-INF\\classes\\ ca.keystore
truststore.password	Specify the pass-phrase you selected for the ca.keystore file.
Debug_ssl	Keep the default (false).

3. Copy and rename the file lwssofmconf.xml.sample (under <symphonyadapter>\WEB-INF\classes) to lwssofmconf.xml.

4. Edit the parameter values in bold in lwssofmconf.xml as shown below.

```
<lwsso-config
  xmlns="http://www.hp.com/astsecurity/idmenablmentfw/lwsso/1.0">
  <webui enabled="true">
       <web-lwsso>
         <lwsso startLWSSO="enabled">
            <domain><Domain name of the Symphony Adapter server
host></domain>
            <crypto cipherType="symmetricBlockCipher"
              engineName="AES" paddingModeName="CBC" keySize="256"
              encodingMode="Base64Url" initString="<Initial string
shared by all applications that support LW-SSO>"></crypto>
            <expirationPeriod>50</expirationPeriod>
         </lwsso>

         <protectedDomains>
           <url><Domain name of another application (for example,
Release Control)'s server host></url>
         </protectedDomains>

         <roleSecurityFrameworkIntegration
           rolePrefix="ROLE_"
           fromLWSSOToSecurityFramework="both"
           fromSecurityFrameworkToLWSSO="enabled"
           caseConversion="upperCase"/>

         <groupSecurityFrameworkIntegration
           fromLWSSOToSecurityFramework="both"
           fromSecurityFrameworkToLWSSO="enabled"
           caseConversion="upperCase"/>
        </web-lwsso>
  </webui>
</lwsso-config>
```

5. Restart Tomcat.

Configuring the Service Manager web client for LW-SSO support

To configure the Service Manager web client for LW-SSO support, you must first configure the Service Manager Web client for Trusted Sign-On and SSL support with the Service Manager Server. This involves generating and deploying certificates and modifying the sm.ini file on the Service Manager server and web.xml on the web client. See the procedures described above.

IMPORATANT NOTE: Make sure that trusted sign-on with SSL is working from the web client before proceeding!

1. Modify the web.xml file in the web client deployment to enable LW-SSO as shown below.

```
<filter>
   <filter-name>LWSSO</filter-name>
   <filter-
class>com.hp.sw.bto.ast.security.lwsso.LWSSOFilter</filter-class>
 </filter>
 <filter-mapping>
   <filter-name>LWSSO</filter-name>
   <url-pattern>/*</url-pattern>
 </filter-mapping>
```

2. Locate the isCustomAuthenticationUsed context-param element in the web client deployment web.xml. The param-value element should be set to false. It should look like the following:

```
<context-param>
      <param-name>isCustomAuthenticationUsed</param-name>
      <param-value>false</param-value>
  </context-param>
```

3. Modify the application-context.xml file located in the WEB-INF\classes folder of the Service Manager web tier deployment.

   a. Locate the filterChainProxy bean element. Add the lwSsoFilter to the value element.

```
 <bean id="filterChainProxy"
class="org.acegisecurity.util.FilterChainProxy">
    <property name="filterInvocationDefinitionSource">
      <value>
        CONVERT_URL_TO_LOWERCASE_BEFORE_COMPARISON
        PATTERN_TYPE_APACHE_ANT

/**=httpSessionContextIntegrationFilter,lwSsoFilter,anonymousPr
ocessingFilter
      </value>
    </property>
 </bean>
```

   b. Uncomment the lwSsoFilter bean and the lwSsoIntegration bean. **Note**: If it is not there, add it.

```
<!-- This bean is used for HP Lightweight Single Sign-on, to
integrate with other Hewlett-Packard software products.
Uncomment it here and reference it in the      filterChainProxy
as commented above. -->
 <bean id="lwSsoFilter"
class="com.hp.ov.sm.client.webtier.lwsso.LwSsoPreAuthentication
Filter">
    <property name="authenticationManager">
      <ref bean="authenticationManager"/>
    </property>
    <property name="defaultRole">
      <value>ROLE_PRE</value>
    </property>
  </bean>
```

```
        <bean id="lwSsoIntegrationBean"
      class="com.hp.ov.sm.client.webtier.lwsso.LwSsoIntegration"/>
```

4. In the lwssofmconf.xml file located in the WEB-INF\classes folder of the Service Manager Web client deployment, enable webui and add domain and crypto settings. The domains listed in the file should reflect your deployment. The initSTring in the crypto element needs to be the same on all applications using LWSSO. The file should look like the following after modification:

```
<?xml version="1.0" encoding="UTF-8"?>
<lwsso-config
  xmlns="http://www.hp.com/astsecurity/idmenablmentfw/lwsso/1.0">
  <webui enabled="true">
    <web-lwsso>
      <lwsso startLWSSO="enabled">
        <domain><Domain name of the Service Manager Web tier server
host> </domain>
        <crypto cipherType="symmetricBlockCipher"
          engineName="AES" paddingModeName="CBC" keySize="256"
          encodingMode="Base64Url" initString="<Initial string shared
by Service Manager and all other applications that support LW-
SSO>"></crypto>
          <expirationPeriod>50</expirationPeriod>
      </lwsso>

      <logoutURLs>
         <url><Service Manager Web tier name, for example,
sm920>/goodbye.jsp</url>
      </logoutURLs>

      <protectedDomains>
        <url><Domain name of the server host of another HP
application that supports LW-SSO. For each protected domain, a
separate url element is required.></url>
      </protectedDomains>

      <roleSecurityFrameworkIntegration
         rolePrefix="ROLE_"
         fromLWSSOToSecurityFramework="external"
         fromSecurityFrameworkToLWSSO="enabled"
         caseConversion="upperCase"/>

      <groupSecurityFrameworkIntegration
         fromLWSSOToSecurityFramework="external"
         fromSecurityFrameworkToLWSSO="enabled"
         caseConversion="upperCase"/>
    </web-lwsso>
   </webui>
</lwsso-config>
```

5. Modify the <Tomcat home directory>\conf\server.xml by adding **tomcatAuthentication="false"** as shown below.

```
<Connector port="8009"
        enableLookups="false" tomcatAuthentication="false" redirectPort="8443"
debug="0"
        protocol="AJP/1.3" />
```

6. Restart your Tomcat server.

August 2010