

# HP Service Manager

Software Version: 9.41

For the supported Windows® and UNIX® operating systems

## Server security help topics for printing

Document Release Date: September 2015  
Software Release Date: September 2015



## Legal Notices

### Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

### Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

### Copyright Notice

© 1994-2015 Hewlett-Packard Development Company, L.P.

### Trademark Notices

Adobe® is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

UNIX® is a registered trademark of The Open Group.

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

For a complete list of open source and third party acknowledgements, visit the HP Software Support Online web site and search for the product manual called HP Service Manager Open Source and Third Party License Agreements.

## Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to: <https://softwaresupport.hp.com/>.

This site requires that you register for an HP Passport and to sign in. To register for an HP Passport ID, click **Register** on the HP Support site or click **Create an Account** on the HP Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

## Support

Visit the HP Software Support site at: <https://softwaresupport.hp.com>.

This website provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support website to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and to sign in. Many also require a support contract. To register for an HP Passport ID, click **Register** on the HP Support site or click **Create an Account** on the HP Passport login page.

To find more information about access levels, go to: <https://softwaresupport.hp.com/web/softwaresupport/access-levels>.

**HPSW Solutions Catalog** accesses the HPSW Integrations and Solutions Catalog portal website. This site enables you to explore HP Product Solutions to meet your business needs, includes a full list of Integrations between HP Products, as well as a listing of ITIL Processes. The URL for this website is <https://softwaresupport.hp.com/group/softwaresupport/search-result/-/facetsearch/document/KM01702710>.

## About this PDF Version of Online Help

This document is a PDF version of the online help. This PDF file is provided so you can easily print multiple topics from the help information or read the online help in PDF format. Because this content was originally created to be viewed as online help in a web browser, some topics may not be formatted properly. Some interactive topics may not

be present in this PDF version. Those topics can be successfully printed from within the online help.

# Contents

<b>System Security</b>	<b>7</b>
<b>Encryption of configuration file settings</b>	<b>9</b>
Encrypt values in the Service Manager configuration file	10
<b>Encryption of operator passwords</b>	<b>12</b>
<b>Encryption of client keystore passwords</b>	<b>13</b>
<b>Inactivity timer</b>	<b>16</b>
Respond to the inactivity warning message	18
<b>Lockout feature</b>	<b>20</b>
Enable the user lockout feature	20
Lock out a user	21
Reset a locked out user	21
View a user's lockout history	22
<b>System quiesce: Login restrictions</b>	<b>23</b>
Enable logging of user access	25
Enable login restrictions	25
Disable login restrictions	25
Enable tracking of operator times	26
<b>Mandanten file security</b>	<b>27</b>
Add a restricting query to a security group	28
Create a security group	29
Define the Mandant field for a security group	30
Enable Mandanten security on a file	31
<b>Multicompany mode</b>	<b>33</b>
Enable multicompany mode	33

Script utilities .....	34
Security tables .....	35
Secure Sockets Layer (SSL) encryption and server certificates .....	36
Secure Sockets Layer (SSL) configuration options .....	37
Requirements for optional SSL encryption .....	38
Example: Viewing the contents of a cacerts file .....	39
Enable SSL encryption for web clients .....	40
Enable SSL encryption for Windows clients .....	41
Update the cacerts keystore file .....	41
Requirements for required SSL encryption .....	42
Example: Enabling required SSL encryption .....	44
Example: Generating a server certificate with OpenSSL .....	45
Example: Viewing the contents of a cacerts file .....	49
Update the cacerts keystore file .....	50
Requirements for required SSL encryption and client authentication .....	50
Example: Enabling required SSL encryption and client authentication .....	53
Example: Generating a client certificate with OpenSSL .....	55
Example: Generating a server certificate with OpenSSL .....	58
Example: Viewing the contents of a cacerts file .....	61
Add a client certificate to the web tier .....	62
Add a client certificate to the Windows client .....	63
Update the cacerts keystore file .....	64
Use keytool to create a certificate request .....	64
Use keytool to create a private key .....	65
Requirements for required SSL encryption and trusted clients .....	66
Example: Enabling required SSL encryption and trusted clients .....	69
Example: Generating a client certificate with OpenSSL .....	72
Example: Generating a server certificate with OpenSSL .....	75
Example: Viewing the contents of a cacerts file .....	78
Add a client certificate to the web tier .....	79
Update the cacerts keystore file .....	80
Use keytool to create a certificate request .....	81
Use keytool to create a private key .....	81
Enable SSL encryption for external Web Services .....	82
Enable SSL encryption for published Web Services .....	84
What are PEM files? .....	85
What is a cacerts file? .....	85

Windows client error: No trusted certificate found .....	85
<b>TLS 1.2 Support and Configuration .....</b>	<b>87</b>
<b>Trusted sign-on .....</b>	<b>88</b>
Requirements for trusted sign-on .....	89
Example: Enabling trusted sign-on .....	90
Example: Generating a server certificate with OpenSSL .....	93
Example: Generating a client certificate with OpenSSL .....	97
Example: Configuring the web server for trusted sign-on .....	99
Example: Viewing the contents of a cacerts file .....	101
Update the cacerts keystore file .....	102
Use keytool to create a certificate request .....	103
Use keytool to create a private key .....	104
<b>Common Access Card (CAC) sign-on .....</b>	<b>105</b>
Requirements for CAC sign-on .....	109
Example: enabling CAC sign-on .....	111
<b>FIPS mode .....</b>	<b>129</b>
Configure FIPS mode in Service Manager .....	131
Server side .....	131
Client side .....	133
<b>Tokenization .....</b>	<b>139</b>
<b>Send Documentation Feedback .....</b>	<b>143</b>

# System Security

HP Service Manager includes a number of security options you can configure after installation. The following table lists the security options you can enable or configure from a new development environment installation.

Feature	Description	Default state in new installations
<a href="#">"Encryption of configuration file settings" on page 9</a>	A security option that protects values listed in the configuration file	Enabled
<a href="#">"Encryption of operator passwords" on page 12</a>	A security option that protects the passwords listed in operator records	Enabled
<a href="#">"Inactivity timer" on page 16</a>	A security option that automatically closes user sessions that have been idle for a specified period of time (except for those operators who are on the exception list)	Enabled
<a href="#">"Lockout feature" on page 20</a>	A security option that automatically disables a user account if the user fails to provide the correct password after a specified number of attempts	Enabled
<a href="#">"Mandanten file security" on page 27</a>	A security option that filters the data that operators can see when they query specific files	Disabled
<a href="#">"Multicompany mode" on page 33</a>	A security option that filters the company information that service desk technicians see when creating service desk interactions and opening incidents	Disabled
<a href="#">"Requirements for optional SSL encryption" on page 38</a>	An implementation option that provides SSL encryption with a sample server certificate. Each Service Manager client can choose to enable or disable the SSL encryption.	Disabled
<a href="#">"Requirements for required SSL encryption" on page 42</a>	An implementation option that requires SSL encryptions for all connections	Disabled
<a href="#">"Requirements for required SSL encryption and client authentication" on page 50</a>	An implementation option that requires SSL encryptions for all connections and validates the client's certificates	Disabled

Feature	Description	Default state in new installations
<a href="#">"Requirements for required SSL encryption and trusted clients" on page 66</a>	An implementation option that requires SSL encryptions for all connections and restricts connections to a list of trusted clients	Disabled
<a href="#">"Script utilities" on page 34</a>	A security option that enables checksum calculation for Service Manager binaries and data security deletion.	Disabled
<a href="#">"Secure Sockets Layer (SSL) encryption and server certificates" on page 36</a>	An implementation option to encrypt communications between your Service Manager server and clients	Disabled
<a href="#">"Trusted sign-on" on page 88</a>	An implementation option that enables HP Service Manager clients to automatically log on using the same authentication information as users entered when they logged onto their client workstation's operating system.	Disabled
<a href="#">"Common Access Card (CAC) sign-on" on page 105</a>	An implementation option that enables HP Service Manager clients to automatically log on using authentication information from the user's personal certificate stored in the user's CAC card.	Disabled
<a href="#">FIPS mode</a>	A security option that requires data be encrypted using the FIPS-compliant AES encryption algorithm.	Disabled
<a href="#">"Tokenization " on page 139</a>	A security option that enables tokenization in the web client to safeguard sensitive data.	<b>Enabled</b>
<a href="#">Enhanced query hash algorithm</a>	A security option that enables the enhanced query hash algorithm.	Disabled



# Encryption of configuration file settings

You can encrypt values within the HP Service Manager configuration file (sm.ini) to protect passwords and authentication information. The encryption scheme is ideal for connection parameters such as RDBMS or LDAP user names and passwords. Using a command line utility you can convert any value to an AES256-256-CBC encrypted value. You can then copy the encrypted value into the configuration file and add an asterisk to the beginning of the parameter name. The asterisk is a required element that indicates to the server that the parameter value is encrypted.

For example, the unencrypted value:

```
sqllogin:rdbmsuser/mypassword
```

Becomes the following value when encrypted:

```
*sqllogin:D51CB23B379C873CBA055FB9A3798375AC93D48BB8AE2CC773D7317E4715EAE7
```

After you have encrypted a configuration file value there is no way to convert it back to clear text. The Service Manager server decrypts the value for administrative purposes, but it does not save or store the decrypted value.

**Note:** Only use the encryption scheme to encrypt server parameters in the Service Manager configuration file (sm.ini). Do not encrypt other values in other files as this may damage your system or cause data loss.

In addition, the values of the following security parameters in the sm.ini file are encrypted automatically during the server startup. You do not need to encrypt them manually.

- changeencrkey
- upgradeencralg
- encryptionkey
- sqllogin
- ldapbindpass
- smtppassword
- svc\_p4passwd
- keystorePass

- truststorePass
- ssl\_trustedClientsPwd

## Encrypt values in the Service Manager configuration file

### Applies to User Roles:

System Administrator

You can use the **sm -encrypt** command to encrypt parameter values in the sm.ini file. You must have administrative access to the server operating system to use this procedure.

**Note:** The values of some security parameters are encrypted automatically during the server startup, including the sqllogin parameter that is used as the example in the following task. For a list of the parameters that are encrypted automatically, see ["Encryption of configuration file settings" on the previous page](#).

To encrypt values in the Service Manager configuration file, follow these steps:

1. Open the HP Service Manager configuration file (sm.ini).
2. Copy the parameter name and parameter value you want to encrypt.  
For example, copy sqllogin:RDBMS user/my password.
3. Open the operating system's command prompt.  
For example, on Windows click **Start > Programs > Accessories > Command Prompt**.
4. Change directories to the RUN folder of your Service Manager installation.  
For example:  

```
cd C:\Program Files\HP\ Service Manager\Server\RUN
```
5. Type the following command:

```
sm -encrypt:<parameter name>:<parameter value>
```

If the string containing the <parameter name>:<parameter value> includes spaces, you must enclose the entire parameter name and value in quotation marks. For example:

```
sm -encrypt:"sqllogin:RDBMS user/my password"
```

**Tip:** The sqllogin credentials are automatically encrypted and added to the sm.ini file if you configure the credentials by running the Server Configuration Utility instead of running this command.

6. Press Enter.

Service Manager displays the new encrypted value.

For example:

```
sqllogin:"RDBMS user/my password" is encrypted:  
99E6136711576164187C6619C6DBA9DFFC89F7EDF186BC5827E3615BE8419CD4
```

7. Copy the new encrypted value.
8. Return to the Service Manager configuration file and add an asterisk to the beginning of the parameter name.

For example:

```
*sqllogin:
```

9. Replace the clear-text value with the encrypted value.

For example:

```
*sqllogin:99E6136711576164187C6619C6DBA9DFFC89F7EDF186BC5827E3615BE8419CD4
```

10. Save the configuration file.

# Encryption of operator passwords

The HP Service Manager server encrypts all operator passwords stored on the database using a SHA512 one-way encryption process that cannot be decrypted. The server automatically encrypts existing passwords on your system the first time they are accessed. You can also do a mass update of the operator table to convert all passwords at once.

You need to update the SQL mapping for the password field to accept a larger character limit. The data policy settings are as follows.

Database object	Requirement
Table	operator
Field	password
Data type	VARCHAR
Size	136 characters

Automatic operator password encryption replaces the legacy data policy encryption option that was controlled by the `encryptionkey` parameter. To convert to the automatic operator password encryption scheme in legacy systems, you must first turn off the existing data policy encryption. The server updates the password encryption the next time it reads the operator record.

Service Manager clients use a two-way encryption process (PBE with MD5, DES in non-FIPS mode, and AES in FIPS mode) to secure operator passwords when communicating with the server. The server decrypts the password sent from the client and then one-way encrypts it to compare the results to the encrypted value stored in the database. The server never stores the operator password in an unencrypted form.

If your Service Manager implementation uses LDAP authentication, the server must still send an unencrypted operator password to the directory service because LDAP servers are unaware of Service Manager's encryption scheme. If you require encryption between Service Manager and the LDAP server, you can configure OpenSSL or another standard encryption scheme between the two servers.

# Encryption of client keystore passwords

Service Manager supports encryption of keystore passwords in the Windows and web clients.

## Windows client

The Windows client keystore password that you enter in **Window > Preferences > HP Service Manager > Security** is automatically encrypted and stored in the following file:

```
<Your user workspace dir  
>\ServiceManager\workspace\.metadata\plugins\org.eclipse.core.runtime\settings\com.hp.ov.sm.client.eclipse.base.prefs
```

### Limitation

If you have installed one or multiple instances of the Windows client earlier than version 9.34.p2 on the same host, after you install the SM 9.41 Windows client on the same host and put all the instances into alternate use, you may have problems with the **Use SSL Encryption** option or the **Use Trusted Sign-on** connection option. Therefore, we strongly recommend you to uninstall all other Windows client instances before you install the Service Manager 9.41 Windows client. If you still have problems with the **Use SSL Encryption** option or the **Use Trusted Sign-on** connection option after you upgrade your Windows client (for example, you may roll back to an earlier version and then upgrade Service Manager to version 9.41), you need to update the com.hp.ov.sm.client.eclipse.base.prefs file. To do so, follow these steps:

1. Open the following preference file with a text editor:  

```
<your home folder  
>\ServiceManager\workspace\.metadata\plugins\org.eclipse.core.runtime\settings\com.hp.ov.sm.client.eclipse.base.prefs
```

For example, your home folder can be C:\Users\*<your username>*.

2. Modify the value of the **safePassword** parameter from **true** to **false**, or delete this parameter.
3. Save and close the file.
4. Restart the Windows client.

5. Navigate to **Window > Preferences > HP Service Manager > Security**, and re-enter the client keystore password.

## Web client

The **keystorePassword** parameter has been removed from the web tier configuration file (web.xml) since version 9.34p2, and you must enter your web client keystore password in a webtier.properties file that is located in the following folder:

<Customize-Folder>/config/webtier.properties (where <Customize-Folder> is the folder specified in the **customize-folder** parameter in the web.xml file.)

**Tip:** You can use the <Webtier>/WEB-INF/webtier.properties file as a template for your <Customize-Folder>/config/webtier.properties file.

When the web application server is started, Service Manager changes this value to an encrypted string. The following steps describe how you set a new or update an existing web client keystore password.

To set a web client keystore password using the <Customize-Folder>, follow these steps:

1. Make sure the **customize-folder** parameter is configured in the web tier configuration file (web.xml).
2. Create a webtier.properties file in the <Customize-Folder>/config directory.

**Note:** Make sure the web application server has read/write access to this directory.

3. Open the file in a text editor, and add the following line to the file:

```
keystorePassword=< your keystore password in clear text format>
```

4. Save and close the file.
5. Restart the web application server.

The password you entered is now encrypted. In the meantime, Service Manager adds the following line to the webtier.properties file:

```
safePassword=true
```

To update an existing keystore password of the web client, follow these steps:

1. Stop the web application server.
2. Open the webtier.properties file with a text editor.
3. Modify the value of the **safePassword** parameter from **true** to **false**, or delete this parameter.
4. Modify the value of the **keystorePassword** parameter to a new password.
5. Save and close the file.
6. Restart the web application server.

## Inactivity timer

The inactivity timer functionality closes user sessions that have been idle for a specified period of time. This feature is enabled by default and the time-out value is 30 minutes. System administrators can configure required settings and start or stop the inactivity timer.

**Note:** You need the `SysAdmin` capability word to configure and start/stop the inactivity timer.

## Configure inactivity timer settings

Administrators can configure required settings at a system level and if needed, also an operator level, as described in the following table.

**Note:** In a horizontal/vertical scaling environment, you need to configure the inactivity timer settings only once (that is, only on any one of your server nodes).

Configuration Level	Settings	Notes
Operator Level	<ul style="list-style-type: none"><li>• <b>Warning Time</b></li><li>• <b>Allowed Inactive Time</b></li></ul>	<ul style="list-style-type: none"><li>• To configure the settings, click the <b>Security</b> tab in an operator record.</li><li>• Operator level settings take precedence over system level settings; if an operator record has empty inactivity timer settings, the system level settings are used for the operator.</li><li>• An allowed inactive time of zero (0) means inactivity session timeout will never happen to the operator.</li><li>• A zero or empty warning time means no warning message will display to the operator.</li></ul>
System Level	<ul style="list-style-type: none"><li>• <b>Number of minutes to allow idle user</b> ("allowed inactive time" )</li><li>• <b>Warn user within &lt;x&gt; minutes of</b></li></ul>	Click <b>System Administration &gt; Ongoing Maintenance &gt; System &gt; Start Inactive Timer &gt; Start Inact.</b>



Configuration Level	Settings	Notes
	<b>termination</b> ("warning time" )  • <b>DO NOT terminate these users</b> ("exception list")	

The following table is an explanation of the settings described above.

Setting	Description
Allowed inactive time	This is a time limit that users can remain inactive until the inactivity timer closes their user session.
Warning time	This is an amount of time between the appearance of a warning message and the termination of the session.  <b>Caution:</b> The warning time should be no greater than the allowed inactive time.
Exception list	This is a list of operators that the inactivity timer ignores. This list should include all operators that should not be logged out regardless of the type of license (Named or Floating license).

## When the inactivity timer settings take effect

Operator level settings take effect the next time the user logs in, while system level settings take effect for all new user sessions.

**Note:** Internet Explorer on a Windows Server operating system that does not support audio cannot display the warning messages. The messages are blocked together with the notification sound.

## Start the inactivity timer

**Note:** Starting with SM9.32, the **inactive.startup** background process has been obsoleted. You no longer need to start this process in order for the inactivity timer functionality to work.

To enable the inactivity timer functionality for users, start the inactivity timer as follows:

1. Click **System Administration > Ongoing Maintenance > System > Start Inactivity Timer**.  
The inactive.setup.g form opens.
2. Type or select the inactivity timer record information. See ["Configure inactivity timer settings" on page 16](#).
3. Click **Start Inact**.

If you want to disable the inactivity timer functionality for all users, you can stop it as follows.

1. Click **System Administration > Ongoing Maintenance > System > Start Inactivity Timer**.

The inactive.setup.g form opens.

2. Click **Stop Inact**.

The **Start Inact** button is displayed.

After modifying your system level inactivity timer settings, you need to restart the inactivity timer as follows.

1. Click **System Administration > Ongoing Maintenance > System > Start Inactivity Timer**.

The inactive.setup.g form opens.

2. Modify the settings as needed.
3. Click **Save**. Service Manager displays the message: The inactive process has been set up to look for users idle after <n> minutes.

## Respond to the inactivity warning message

### Applies to User Roles:

All users

You may receive an inactivity warning message depending on the inactivity timer settings your administrator has configured. When this happens, you can respond as needed to either time out or extend your session.

The following table describes your actions in the Windows/web client and the corresponding inactivity timer behavior.

Client	Warning Message	Action
Web	A warning message displays in a pop-up window.	Click <b>OK</b> : Your session extends and the inactivity timer restarts counting. Click <b>Cancel</b> , press <b>ESC</b> , or make no response: The inactivity timer continues the counting and terminates your session in the end.
Windows	A warning message displays in the toolbar.	Perform any operation: Your session extends and the inactivity timer restarts counting.  Make no response: The inactivity timer continues the counting and terminates your session in the end.

# Lockout feature

The lockout feature automatically disables a user account if the user fails to provide the correct password after a specified number of attempts. Your user profile must include the system administration capability word to use this feature.

## Enable the user lockout feature

### Applies to User Roles:

System Administrator

The user lockout feature automatically disables a user account if the user fails to provide the correct password after a specified number of attempts.

**Note:** Enabling the user lockout feature is the only way to restrict access from the Windows client. The **Attempts Per Login Session** settings does not affect Windows clients because the Windows client creates a new session for each login attempt.

To enable the user lockout feature:

1. Click **System Administration > Base System Configuration > Miscellaneous > System Information Record**.
2. Click the **Logon Info** tab.
3. In the Use User Lockout section, select **Use User Lockout**.
4. In the **Attempts Until Lockout** field, type the number of login attempts the user has until HP Service Manager locks out the account.
5. Select one of the following options:
  - **Complete Lockout** — select this option to deny the user access to Service Manager until a system administrator resets the lockout.
  - **Lock Out Time Period** — select this option to deny the user access to Service Manager for a specified period of time.  
Type the time period in the following format: *Dayshours:minutes:seconds*. For example, 4 03:02:01 locks out the user for 4 days, 3 hours, 2 minutes, and 1 second.

6. Click **Save**.

## Lock out a user

### **Applies to User Roles:**

System Administrator

To lock out a user:

1. Click **System Administration > Ongoing Maintenance > Operators**.
2. Type or select optional search criteria.
3. Click **Search**.
4. Select the operator to lock out.
5. Click the **Security** tab.
6. Select the **Administrative Lockout** option
7. Click **Save**.

## Reset a locked out user

### **Applies to User Roles:**

System Administrator

To reset a locked out user:

1. Click **System Administration > Ongoing Maintenance > Operators**.
2. Type or select optional search criteria.
3. Click **Search**.
4. Select the operator record.
5. Click the **Security** tab.
6. Click **More** or the More Actions icon, and then select **User Lockout Reset**.

7. Click **Save**.

HP Service Manager unlocks the user's operator record and clears the **Failed Login Count**, **Locked Until**, and **User has been Locked?** fields.

## View a user's lockout history

### **Applies to User Roles:**

System Administrator

To view a user's lockout history

1. Click **System Administration > Ongoing Maintenance > Operators**.
2. Click **Search**.
3. Select the operator record to view.
4. Click the **Security** tab.

## System quiesce: Login restrictions

Quiesce mode sets login restrictions to prevent users from logging on to Service Manager processes. This gives System Administrators a way to stop users from logging on to Service Manager processes, and wait for existing users to gracefully log off before starting system maintenance or testing tailoring activities.

There are three levels of login restrictions. Quiesce level 1 restricts all users, except System Administrators, from logging on to Service Manager processes. Quiesce level 2 restricts all users from logging on to Service Manager processes. Quiesce level 0 (zero) sets Service Manager processes to allow user logins. In the vertical scaled or horizontal scaled environment, Service Manager load balancer does not forward any client connection requests to Service Manager processes that are in quiesce level 1 or 2. If System Administrators want to connect to a Service Manager process in quiesce level 1, they must connect directly to the Service Manager process without connecting through Service Manager load balancer.

Quiesce mode information is stored in the shared memory. When a Service Manager process sets a quiesce level in the shared memory, all other Service Manager processes that read from the same shared memory have the same quiesce level. So if one Service Manager process sets a quiesce level on a host, all Service Manager processes have the same level of quiesce on that host. In a horizontal scaled environment, the "-host:<host name or IP>" or "-group" option can be used with the "sm - quiesce:<quiesce level>" command. The "sm -quiesce:<1 or 2 or 0> -group" command sets the quiesce level to all Service Manager processes on all hosts within the horizontal scaled group.

In a vertical scaled environment when Service Manager processes are running and you issue the "sm - quiesce:1" or "sm -quiesce:2" command, all Service Manager processes on the local host are set to quiesce level 1 or quiesce level 2, respectively. The Service Manager load balancer stops forwarding any new client connection requests to the Service Manager processes, since they are quiesced. If any user tries to connect to Service Manager load balancer at this point, the user receives a message that states "max session exceed" from the load balancer, as there are no available Service Manager processes. Existing users on the system are not affected. However, once existing users log off, they cannot log back on until after the System Administrator changes the quiesce level back to 0 (zero). Once all users have logged off the system, the System Administrator can perform system maintenance. When system maintenance is complete, the System Administrator can issue the "sm -quiesce:0" command to set the quiesce level back to 0, so there are no login restrictions to all Service Manager processes. All Service Manager processes now accept user logins and Service Manager load balancer forwards client connection requests to these Service Manager processes.

In a horizontal scaled environment, assume there are two hosts in the horizontal scaled group, Host A and Host B. While the system is running, System Administrators can quiesce Host A for maintenance and

keep Host B running by issuing the command `"sm -quiesce:<1 or 2> -host:<Host A name or IP address>"` on either Host A or Host B. Service Manager load balancer then forwards all client requests to Host B, as all Service Manager processes on Host A are quiesced. Existing users on Host A are not affected until they log off. If users try to log back on, Service Manager load balancer redirects their connection requests to Host B. After maintenance is complete on Host A, the System Administrator issues the command `"sm -quiesce:0 -host:<Host A name or IP address>"` to bring Host A back to service. This way the System Administrator can maintain one of the hosts in a horizontal scaled group and avoid down time. The System Administrator can also quiesce all Service Manager processes in the group by issuing the `"sm -quiesce:<1 or 2> -group"` command. When the maintenance is complete, the System Administrator can then set all Service Manager processes in the group back to non-quiesce mode by issuing the `"sm -quiesce:0 -group"` command.

A System Administrator can restrict logins to Service Manager using the `system.quiesce` application. System maintenance tasks include the following:

- Upgrading from one version of Service Manager to another
- Tailoring forms, tables, or format controls

The `system.quiesce` application provides three levels of login restrictions:

Restriction level	Description
<b>Level 0</b>	Service Manager has no login restrictions and accepts all logins normally.
<b>Level 1</b>	<p>Service Manager restricts login to operators who have the SysAdmin capability word. Service Manager denies login to all other operators and displays the message:</p> <pre>System quiesced, you cannot login at this time.</pre> <p>When you issue <code>sm -quiesce:1 -group</code> on a primary or secondary host, all Service Manager processes (except Service Manager load balancer) are set to quiesce mode 1. Service Manager load balancer updates its available node list to 0. If users try to connect to Service Manager load balancer at this time, they receive the following message, since there are no available Service Manager processes.</p> <pre>Max session exceeded.</pre>
<b>Level 2</b>	<p>Service Manager denies login to all operators and displays the message:</p> <pre>System quiesced, login restricted at this time.</pre>

**Note:** A quiesced system restricts new login attempts only. Currently logged on users can continue working until they log off.



## Enable logging of user access

### Applies to User Roles:

System Administrator

If you enable system logging of user access, HP Service Manager records the time and user ID each time an operator logs on or logs off.

To enable logging of user access:

1. Click **System Administration > Base System Configuration > Miscellaneous > System Information Record**.
2. On the General tab, select the **Syslog Audit** option.
3. Click **Save**.

## Enable login restrictions

### Applies to User Roles:

System Administrator

To enable login restrictions:

1. Click **System Administration > Ongoing Maintenance > System > Connection Restrictions**.
2. Click one of the following options:
  - **Set Level 0** — No login restrictions
  - **Set Level 1** — Only system administrators can log in
  - **Set Level 2** — No operators can loginHP Service Manager displays the current access level.

## Disable login restrictions

### Applies to User Roles:

System Administrator

To disable login restrictions:

1. Click **System Administration > Ongoing Maintenance > System > Connection Restrictions**.
2. Click **Set Level 0**.

## Enable tracking of operator times

### **Applies to User Roles:**

System Administrator

You can track how long each operator edits an Incident record by enabling a tracking option in the Incident Management environment.

To enable tracking of operator times:

1. Click **System Administration > Ongoing Maintenance > Environment Records > Incident Management Environment**.
2. Select the **Track Operator Times?** checkbox.
3. Click **Save**.

## Mandanten file security

Mandanten is an optional file security feature that filters the data that operators can see when they query specific files. Rather than having access to all the records in a file, operators who are members of a security group see only the records that meet the specific filtering criteria of their group. The system administrator defines the filtering conditions when creating a security group. The system administrator decides which operators belong to particular security groups and can assign operators to any number of security groups. Operators who are members of multiple security groups see only the records that match all their separate filtering conditions.

At login, HP Service Manager reads the operator record to determine the security groups of which the operator is a member and uses this information to determine the files to which the operator has limited access. When an operator queries a restricted file, Service Manager reads the security group records to determine the filtering conditions to apply to the query. Service Manager then returns only those records that match the filtering conditions in the security group records.

Unlike Format Control, which provides security at the application layer, Mandanten secures files at the database layer. Any file that a system administrator restricts from an operator with Mandanten always uses the filtering conditions regardless of the operator's user role and application profile. Only operators who are not members of any security group can have unrestricted access to files protected by Mandanten.

Typically, a system administrator enables Mandanten file security in a multi-company environment where each company wants to ensure that only their users see the data relevant to their business. However, system administrators can also use Mandanten to conceal department records selectively within an organization. For example, a system administrator could create two filtering conditions for an operations and finance department that allow the operations personnel to see their own Incident records and devices but not those belonging to the finance department.

## Setting filtering conditions

To enable Mandanten file security, a system administrator must create records in two files:

- **scsecuritygroup** — The system administrator uses this file to define the security group name and the field values to be used as the filtering condition.
- **scmandant** — The system administrator uses this file to define the Service Manager file to be protected and the field to be read for the filtering condition. The field defined in this file is referred to as the Mandant field. You can only define one Mandant field for each file you want to protect,

although the Mandant field can contain an array of values. The Mandant field you chose must be defined either in the file you want to protect or by a virtual join in another file.

System administrators can define additional filtering conditions on fields other than the Mandant field. These additional filtering conditions are referred to as restricting queries because they further restrict the data that an operator can access. To define a restricting query, a system administrator must create a record in the **scaccess** file.

## Restrictions

You cannot enable Mandanten file protection on the following shared system files:

- code
- datadict
- dbdict
- environment
- format
- formatctrl
- link
- menu
- operator
- tzfile

## Add a restricting query to a security group

### **Applies to User Roles:**

System Administrator

You must have the **SysAdmin** capability word to use this procedure.

You can add a restricting query to a security group to further limit the data that operators can see when they query a Mandanten protected file.

To add a restricting query to a security group:

1. Click **Tailoring > Database Manager**.  
The Database Manager form (format.prompt.db.g) opens.
2. In the **Form** field, type **scaccess**.
3. Click **Search**.  
The Mandanten Restricting Query form (scaccess.g) opens.
4. In the **File Name** field, type or select the name of the Mandanten protected file you want to further restrict.
5. In the **Security Group ID** field, type the name of the security group that you want to further restrict.
6. In the **Restricting Query** field, type the SQL query you want to use to further restrict the data operators can access  
  
**Note:** Typically, this query is of the format *field="value"*.
7. Click **Add**.  
HP Service Manager displays the message:  
scaccess record added.

## Create a security group

### Applies to User Roles:

System Administrator

You must have the **SysAdmin** capability word to use this procedure.

You must create a security group to enable the Mandanten file security feature.

To create a security group:

1. Click **Tailoring > Database Manager**.  
The Database Manager form (format.prompt.db.g) opens.
2. In the **Form** field, type **scsecurity**.
3. Click **Search**.  
The Mandanten Security Groups form (scsecuritygroup.g) opens.

4. In the **Security ID** field, type the name of your new security group.
5. In the **Include Value List** array, type the Mandanten field values that you want to use to grant access to the security group.
6. In the **Exclude Value List** array, type the Mandanten field values that you want to use to deny access to the security group.
7. Click **Add**.  
Service Manager displays the message:  
scsecuritygroup record added.

## Define the Mandant field for a security group

### Applies to User Roles:

System Administrator

You must have the **SysAdmin** capability word to use this procedure.

You must define the Mandant field to enable the Mandanten file security feature.

To define the Mandant field for a security group:

1. Click **Tailoring > Database Manager**.  
The Database Manager form (format.prompt.db.g) opens.
  2. In the **Form** field, type **scmandant.g**.
  3. Click **Search**.  
The Mandanten Field Restriction form (scmandant.g) opens.
  4. In the **File Name** field, type or select the name of the file you want to protect.
  5. In the **Mandant Field Name** field, type the name of the field you want HP Service Manager to read when it checks whether an operator meets the filtering conditions.
- Note:** This field must either be defined in the file you select or be defined in another file as a virtual join.

6. If the Mandant field is a virtual join field, type the following additional information:
  - In the **Linkage Field Name** field, type the name of the field that stores the virtual join reference.
  - In the **Source File Name** field, type the name of the external file that contains the Mandant field.
  - In the **Source Field Name** field, type the name of the external field that stores the Mandant field values you want Service Manager to read when it checks whether an operator meets the filtering conditions.

**Note:** If the source field contains an array of values, then Service Manager will search the array for valid filtering conditions. If any one of the filtering conditions is met, then the operator will have access to the file.

- In the **Exclude Field** field, type any values that you want Service Manager to exclude from queries against the Mandant field.

7. Click **Add**.

Service Manager displays the message:  
scmandant record added.

## Enable Mandanten security on a file

### Applies to User Roles:

System Administrator

You must have the **SysAdmin** capability word to use this procedure.

To enable Mandanten security on a file:

1. Create a security group to define the filtering conditions you want to use on the protected file.
2. Define the Mandant field that you want HP Service Manager to read when it checks whether an operator meets the filtering conditions.
3. Add any restricting queries you want to add to the security group.  
Restricting queries further restricts the data that operators in the security group can access.
4. Assign operators to security groups that match the data you want them to access.

**Note:** If the system administrator does not assign an operator to a security group, then that operator can see an unfiltered view of all the files to which the operator's user role and application profile permit access.



# Multicompany mode

You can run HP Service Manager in multicompany mode to filter the company information that service desk technicians see when creating service desk interactions and opening incidents. Normally, support desk technicians can see information about all the companies that the service desk supports. However, when a system administrator enables multi-company mode, service desk technicians only see information relevant to the company they are currently supporting.

For additional data security, you can enable the Mandanten feature.

## Enable multicompany mode

### Applies to User Roles:

System Administrator

You must have the **SysAdmin** capability word to use this procedure.

To enable multicompany mode:

1. Click **System Administration > Base System Configuration > Miscellaneous > System Information Record**.
2. On the General tab, select the **Run in Multi-Company Mode** option.
3. Click **Save**.

HP Service Manager displays the message:  
Information record updated.

## Script utilities

Service Manager includes two new scripts:

- `smfsdel.bat`

Provides a mechanism to securely delete files from the file system on which you have installed Service Manager.

- `smfchecksum.bat`

Provides a mechanism to verify the Service Manager binaries.

These scripts were created to meet the Federal Service for Technical and Export Control (FSTEC) certification as per the government regulations of Russia. These two scripts are packaged together with Service Manager. Most customers will have no reason to run these scripts but you can run them freely as you like.

For more information about these two scripts and how to use them, see [Scripts for FSTEC Certification on Service Manager](#).

## Security tables

HP Service Manager applications contain built-in security that enables an administrator to define the rights for individual users (operators). For example, some users may have the right to open a call report or incident, but not to close a call report or incident.

# Secure Sockets Layer (SSL) encryption and server certificates

HP Service Manager supports Secure Hypertext Transfer Protocol (HTTPS), which encrypts and decrypts message requests and responses. Service Manager uses Secure Sockets Layer (SSL) for encryption only and relies on the server to authenticate each operator's user name and password. Service Manager supports SSL for the following connections:

- SSL on the *Service Manager server* to encrypt all communications between clients and the server.
- SSL on *Service Manager clients* to verify the client's identity and limit server connections to these identified clients

## Enabling SSL on the Service Manager server

The primary reason to enable SSL on the Service Manager server is to protect operator user names and passwords that Service Manager clients send with each request as part of an HTTP Basic Authorization header. You can enable SSL on the Service Manager server but not require each client to present an individual client certificate. When you enable SSL on the server only, clients connect to the server using anonymous SSL.

## Enabling SSL on Service Manager clients

The primary reason to enable SSL on Service Manager clients is to restrict access to the server to only those clients known and identified by the server. Enabling client-side SSL requires creating or purchasing signed certificates for each Service Manager client. The Service Manager Web Tier can share a single signed certificate for all Web Client connections. If you enable client-side SSL, HP recommends you also enable server-SSL to encrypt all communications between clients and the server.

## The client/server SSL handshake process

During the client/server handshake process, the client looks at the server certificate, determines which certificate authority signed the certificate, and compares the certificate signature to a list of trusted certificate authorities identified in the `cacerts` file. Service Manager includes a sample server

certificate signed by a fictitious certificate authority and also includes a modified cacerts file that includes the certificate for the fictitious certificate authority.

The client also compares the IP address or host name of the server to the address encrypted in the server certificate. If they do not match, an alert appears and the user can stop the connection. When you start a new installation of Service Manager, it suppresses the alerts. To ensure a secure environment, remove the sample server certificate, install an actual certificate, and modify the cacerts file to list the appropriate certificate authority.

## Secure Sockets Layer (SSL) configuration options

HP Service Manager supports several different Secure Sockets Layer (SSL) configurations that you can use to protect communications between the server and clients. The following is a list of the most common SSL configurations in order of increasing security and setup required:

Configuration Type	SSL encryption?	Server credentials authenticated?	Client credentials authenticated?	Client can bypass login screen?	Additional certificates required
<b>Optional SSL encryption</b>	Optional Set from client	No	No	No	None
<b>Required SSL encryption</b>	Required Set from server	Yes	No	No	<ul style="list-style-type: none"> <li>• Certificate authority certificate</li> <li>• Server certificate</li> </ul>
<b>Required SSL encryption and client authentication</b>	Required Set from server	Yes	Yes	No	<ul style="list-style-type: none"> <li>• Certificate authority certificate</li> <li>• Server certificate</li> <li>• Web tier certificate</li> <li>• Windows client certificates</li> </ul>
<b>Required SSL encryption and trusted clients</b>	Required Set from server	Yes	Yes	No	<ul style="list-style-type: none"> <li>• Certificate authority certificate</li> <li>• Server certificate</li> </ul>

Configuration Type	SSL encryption?	Server credentials authenticated?	Client credentials authenticated?	Client can bypass login screen?	Additional certificates required
					<ul style="list-style-type: none"><li>• Web tier certificate</li><li>• Windows client certificates</li></ul>

## Requirements for optional SSL encryption

This configuration is intended for customers who:

- Do not want to create and maintain their own certificates
- Do not want to take additional measures to protect against complex SSL-related attacks
- Want to choose whether to enable SSL encryption from the client

**Tip:** Administrators can automatically enable the SSL encryption option for all web clients from the **web.xml** file, and from all Windows clients by deploying a customized client.

### Certificates required

No additional certificates are required. HP Service Manager provides the following certificates for SSL encryption out-of-the-box.

- Keystore containing the certificate authority certificate – the default keystore is located in *<Windows client installation path>\plugins\com.hp.ov.sm.client.common\_x.xx*
- Service Manager server certificate – the default certificate is located in *<HP Service Manager server installation folder>\RUN\smsrv.keystore*

### Private keys required

No additional private keys are required. HP Service Manager provides the following private key for SSL encryption out-of-the-box.

- HP Service Manager server Private key – the default private key is located in *<Service Manager server installation folder>\RUN\smsrv.keystore*

### Parameters required in the server configuration file (sm.ini)

- No additional initialization parameters are required when using optional SSL encryption.
- Optional SSL encryption does not require enabling SSL:1.

**Note:** If you are using optional SSL encryption, no additional initialization parameters are required. Once you enable SSL encryption, it is no longer optional.

#### Parameters required in the web tier configuration file (web.xml)

You can set the following web parameter to enable SSL encryption for web clients.

- `ssl – true`

#### Windows client preferences required

You can set the following preference from the **Connections > Advanced** menu.

- Use SSL Encryption – Encrypts communications from Windows client using the server's certificate

If you enable SSL encryption, then you must set the following preference from the **Window > Preferences > HP Service Manager > Security** menu.

- CA certificates file – identify the keystore containing the server's certificate authority certificate. The default keystore file is located in `<Windows client installation path>\plugins\com.hp.ov.sm.client.common_x.xx`

#### Other requirements

None

## Example: Viewing the contents of a cacerts file

To view the entries in a `cacerts` file, you can use the `keytool` utility provided with Sun J2SDK versions 1.4 or later. The following example uses the `-list` command to display the CA certificates in the `cacerts` file.

```
C:\j2sdk1.4.2_04\jre\bin>keytool -list -keystore ./cacerts
Enter keystore password:  changeit
```

```
Keystore type: jks
Keystore provider: SUN
```

Your keystore contains 15 entries

```
thawtepersonalfreemailca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): 1E:74:C3:86:3C:0C:35:C5:3E:C2:7F:EF:3C:AA:3C:D9
thawtepersonalbasicca, Feb 12, 1999, trustedCertEntry,
```

Certificate fingerprint (MD5): E6:0B:D2:C9:CA:2D:88:DB:1A:71:0E:4B:78:EB:02:41  
baltimorecodesigningca, May 10, 2002, trustedCertEntry,  
Certificate fingerprint (MD5): 90:F5:28:49:56:D1:5D:2C:B0:53:D4:4B:EF:6F:90:22  
verisignclass3ca, Jun 29, 1998, trustedCertEntry,  
Certificate fingerprint (MD5): 78:2A:02:DF:DB:2E:14:D5:A7:5F:0A:DF:B6:8E:9C:5D  
gtecybertrustglobalca, May 10, 2002, trustedCertEntry,  
Certificate fingerprint (MD5): CA:3D:D3:68:F1:03:5C:D0:32:FA:B8:2B:59:E8:5A:DB  
thawtepersonalpremiumca, Feb 12, 1999, trustedCertEntry,  
Certificate fingerprint (MD5): 3A:B2:DE:22:9A:20:93:49:F9:ED:C8:D2:8A:E7:68:0D  
thawteserverca, Feb 12, 1999, trustedCertEntry,  
Certificate fingerprint (MD5): C5:70:C4:A2:ED:53:78:0C:C8:10:53:81:64:CB:D0:1D  
verisignclass4ca, Jun 29, 1998, trustedCertEntry,  
Certificate fingerprint (MD5): 1B:D1:AD:17:8B:7F:22:13:24:F5:26:E2:5D:4E:B9:10  
baltimorecybertrustca, May 10, 2002, trustedCertEntry,  
Certificate fingerprint (MD5): AC:B6:94:A5:9C:17:E0:D7:91:52:9B:B1:97:06:A6:E4  
verisignclass1ca, Jun 29, 1998, trustedCertEntry,  
Certificate fingerprint (MD5): 51:86:E8:1F:BC:B1:C3:71:B5:18:10:DB:5F:DC:F6:20  
verisignserverca, Jun 29, 1998, trustedCertEntry,  
Certificate fingerprint (MD5): 74:7B:82:03:43:F0:00:9E:6B:B3:EC:47:BF:85:A5:93  
thawtepremiumserverca, Feb 12, 1999, trustedCertEntry,  
Certificate fingerprint (MD5): 06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A  
gtecybertrustca, May 10, 2002, trustedCertEntry,  
Certificate fingerprint (MD5): C4:D7:F0:B2:A3:C5:7D:61:67:F0:04:CD:43:D3:BA:58  
gtecybertrust5ca, May 10, 2002, trustedCertEntry,  
Certificate fingerprint (MD5): 7D:6C:86:E4:FC:4D:D1:0B:00:BA:22:BB:4E:7C:6A:8E  
verisignclass2ca, Jun 29, 1998, trustedCertEntry,  
Certificate fingerprint (MD5): EC:40:7D:2B:76:52:67:05:2C:EA:F2:3A:4F:65:F0:D8

## Enable SSL encryption for web clients

You can configure your web clients to use the HP Service Manager server's sample certificate for SSL encryption. If The HP Service Manager server requires SSL connections from clients, then this setting is unnecessary because all clients will use the server's signed certificate for SSL encryption. If SSL connections are optional however, then you can use these steps to enable SSL encryption between the client and server without installing an additional signed server certificate.

1. Log on to the web tier system.
2. Stop the web tier web application server.
3. Open the web configuration file (`web.xml`) in a text editor.
4. Set the **ssl** parameter to **true**.



5. Save your changes.
6. Restart your web tier Web application server.

## Enable SSL encryption for Windows clients

You can configure your Windows clients to use the HP Service Manager server's certificate for SSL encryption. If The HP Service Manager server requires SSL connections from clients, then this setting is unnecessary because all clients will use the server's certificate for SSL encryption. If SSL connections are optional, then you can use these steps to enable SSL encryption between the client and server.

1. If you are managing a private certificate authority, import the certificate for your private certificate authority into the `cacerts` keystore.  
If you are using the services of a certificate authority vendor, the `cacerts` file should already contain your vendor's certificate. Open the `cacerts` file and verify that your vendor's certificate is present.
2. Copy the updated `cacerts` keystore to one of two locations:
  - `<Windows client installation path>\plugins\com.hp.ov.sm.client.common_x.xx` folder
  - A network share accessible to your Windows clients
3. Open the HP Service Manager Windows client.
4. Click **Window > Preferences > HP Service Manager > Security**.
5. Click **Browse** for the **CA certificates file** field.
6. Browse to the path of the `cacerts` keystore.
7. Click **OK** to accept the path.
8. Click **OK** to save your changes.
9. Use the Client Configuration Utility to save the custom SSL configuration of this Windows client and deploy the new Windows Client installer to your network.

## Update the `cacerts` keystore file

### **Applies to User Roles:**

System Administrator

If you use a private certificate authority to generate an SSL server certificate, you can add the private certificate authority to the list of trusted certificate authorities that exist in the Java cacerts keystore file. Sun distributes this file with JSSE and with JDK version 1.4.x and later releases. You can then distribute this updated cacerts file to your HP Service Manager clients so that they can validate the server's signed certificate.

**Note:** This procedure requires that you install a Java SDK of version 1.4.x or later on the server where you installed your private certificate authority.

To update the cacerts keystore file:

1. Log on to server where you installed your private certificate authority.
2. Open the operating systems command prompt.
3. Change directories to the Java SDK bin folder.
4. Type the following command to import your private certificate authority's certificate (for example, cacert.pem) into the Java cacerts file that you publish to the rest of your network. Change the path and variables as necessary.  

```
keytool -import -keystore ./cacerts -trustcacerts -file cacert.pem -storepass changeit
```
5. When keytool prompts you, type y to trust the private certificate authority's certificate.

## Requirements for required SSL encryption

This configuration is intended for customers who:

- Want to require SSL encryption for all connections
- Want to protect against complex SSL-related attacks
- Want to authenticate that the HP Service Manager server is a valid host

### Certificates required

You must create or obtain the following certificates for SSL encryption.

- Certificate authority certificate
- Keystore containing the certificate authority's certificate
- HP Service Manager server certificate

### **Private keys required**

You must create or obtain the following private keys for SSL encryption.

- Certificate authority's private key \*
- HP Service Manager server private key

\* This key is only necessary if you are managing your own private certificate authority.

### **Parameters required in the server configuration file (sm.ini)**

- keystoreFile – identify the keystore file containing the HP Service Manager server's certificate and private key
- keystorePass – identify the password to the keystore file containing the HP Service Manager server's certificate and private key
- ssl:1
- sslConnector:1
- truststoreFile – identify the keystore file containing the certificate authority's certificate
- truststorePass – identify the password to the keystore file containing the certificate authority's certificate

### **Parameters required in the web tier configuration file (web.xml)**

You must set the following web parameter.

- cacerts – identify the keystore file containing the certificate authority that signed the server's certificate

### **Windows client preferences required**

You must set the following preference from the **Window > Preferences > HP Service Manager > Security** menu.

- CA certificates file – identify the keystore file containing the certificate authority that signed the server's certificate

### **Other requirements**

You must do the following additional steps to ensure that HP Service Manager can use your private certificates.

- Add your private certificate authority's certificate to a keystore that your web and Windows clients can access
- Ensure that the HP Service Manager server's host name matches the common name (CN) listed in the server's signed certificate

## Example: Enabling required SSL encryption

The following example describes the following SSL configuration.

- Requiring SSL encryption using the HP Service Manager server's signed certificate

**Note:** This example builds on information presented in the topic *Example: Generating a server certificate with OpenSSL*.

1. Generate a signed server certificate for the Service Manager server.
2. Install the server's signed certificate and supporting key.  
Copy the following keystore files into the RUN folder of the Service Manager server.
  - `servercert.keystore` – This keystore file contains the Service Manager server's signed certificate and private key
  - `cacerts` – This keystore file contains the certificate and private key of the certificate authority that signed the server's certificate
3. Stop the Service Manager server.
4. Open the Service Manager initialization file (`sm.ini`) with a text editor.
5. Add the following parameters to require SSL encryption using the Service Manager server's signed certificate.
  - `keystoreFile:servercert.keystore` – identifies the keystore file containing the Service Manager server's certificate and private key
  - `keystorePass:changeit` – identifies the password to the keystore file containing the Service Manager server's certificate and private key
  - `truststoreFile:cacerts` – identifies the keystore file containing the certificate authority's certificate
  - `truststorePass:changeit` – identifies the password to the keystore file containing the certificate authority's certificate

- `ssl:1` – Requires SSL encryption using the server's signed certificate.
  - `sslConnector:1` – requires Service Manager clients to use an HTTPS port when communicating with the server.
6. Save the Service Manager initialization file.
  7. Restart the Service Manager server.
  8. Stop the web application server running the web tier.
  9. Install the certificate authority's certificate on your Service Manager clients.  
Copy the `cacerts` keystore file into the `WEB-INF` folder of the Web application server running the Service Manager web tier. For example:  
  
`C:\apache-tomcat-5.5.17\webapps\sm\WEB-INF`
  10. Configure Service Manager web clients to validate the Service Manager server's signed certificate.  
Open the web configuration file (`web.xml`) in a text editor, and do the following:
    - Set `cacerts` to the `cacerts` file you copied to the `WEB-INF` folder.
  11. Configure Service Manager Windows clients to validate the Service Manager server's signed certificate.  
Click **Window > Preferences > Service Manager > Security**, and do the following:
    - Set **CA Certificates File** to the `cacerts` you copied to the `<Windows client installation path>\plugins\com.hp.ov.sm.client.common_x.xx` folder.

## Example: Generating a server certificate with OpenSSL

The following example describes how to create a signed server certificate using the OpenSSL toolkit as a private certificate authority. This example also uses the `keytool` utility available with the Sun Microsystems™ standard Java Development Kit. You can use a server certificate to enable SSL encryption from the HP Service Manager server.

**Note:** The information contained in this example regarding OpenSSL technology is provided by HP as a courtesy to our customers and partners. This documentation does not replace an OpenSSL reference, and HP encourages you to conduct additional research regarding OpenSSL technology by consulting with sources outside of this document. HP hereby disclaims all liability associated with the use and accuracy of this information. As OpenSSL technology evolves, HP may or may not update this reference.

1. Obtain and install a Java platform on the server you want to use as your private certificate authority. See the Sun Microsystems™ Java Technology site for the necessary software and documentation.
2. Obtain and install OpenSSL on the server you want to use as your private certificate authority. See the OpenSSL Web site for the necessary software and documentation.

**Caution:** HP strongly recommends that you do not install your private certificate authority on the same server as your Service Manager production server.

3. Create and configure an `openssl.conf` file in the `bin` folder of your OpenSSL installation.
4. Open the operating system's command prompt on the private certificate authority server.
5. Change directories to the OpenSSL `bin` folder.
6. Type the following command to create the private key for your private certificate authority:  
`openssl genrsa -des3 -out cakey.pem 2048`
7. When OpenSSL prompts you, type the password phrase you want to use to protect your certificate authority's private key file (`cakey.pem`). For example, `CAKeyPassword`.  
You must use the same password phrase each time you sign a certificate request with your private certificate authority. If you forget this password, you must repeat the steps to create another certificate authority private key.
8. Type the following command to create a public certificate for your private certificate authority:  
`openssl req -new -key cakey.pem -x509 -days 1095 -out mycacert.pem -config .\openssl.conf`
9. Change directories to the Java platform's `bin` folder.
10. Type the following command to import your private certificate authority's certificate into the Java `cacerts` file that you will publish to the rest of your network.  
`keytool -import -keystore ./cacerts -trustcacerts -file mycacert.pem -storepass changeit`
11. When `keytool` prompts you, type `y` to trust the private certificate authority's certificate.

12. Install the updated Java cacerts file on the Service Manager server. Copy the cacerts file to the RUN folder of the Service Manager server. You can also copy the cacerts file to the JRE\lib\security folder of your Java Run Time Environment (RTE) or Java Development Kit (JDK).
13. Install the updated Java cacerts file on the Service Manager web tier. Copy the cacerts file to the *<web application server installation path>\WEB-INF* folder of the Service Manager web tier.
14. Install the updated Java cacerts file on the Service Manager Windows client. Copy the cacerts file to the *<Windows client installation path>\plugins\com.hp.ov.sm.client.common\_x.xx* folder of the Service Manager Windows client.
15. Change directories to the Java platform's bin folder.
16. Type the following command to create a private key and keystore for your Service Manager server.  
`keytool -genkey -alias myserver -keystore servercert.keystore`
17. When keytool prompts you, type the password phrase you want to use to protect your Service Manager server's keystore file. For example, `SMServerKeyPassword`.

**Note:** Do not use the same password as your private certificate authority key password.

18. When keytool prompts you for your first and last name, type the fully qualified host name of your Service Manager server system.

**Note:** This is the same name your clients will use to connect to the Service Manager server.

19. When keytool prompts you for the organization unit, organization, city or locality, state or province, and two-letter country code, type the identification information for your company.
20. Verify the information you provided and type `yes` if it is correct.
21. When keytool prompts you for the password phrase to use for your Service Manager server's private key, press `ENTER` to use the same password as you created for the keystore.

**Note:** The password for the private key must match the password for the keystore file.

22. Type the following command to create a certificate request for your Service Manager server. For example, to create a certificate request for your HP Service Manager server type:  
  

```
keytool -certreq -alias myserver -keystore servercert.keystore -file smserver_certrequest.crs
```
23. When keytool prompts you, type the password for the HP Service Manager server's keystore file (from step 17). For example, `SMServerKeyPassword`.
24. Copy the Service Manager server's certificate request (For example, `smserver_certrequest.crs`) to the OpenSSL bin folder.

25. Change directories to the OpenSSL bin folder.

26. Type the following command to sign the Service Manager server's certificate request with your private certificate authority:  
  

```
openssl x509 -req -days 365 -in smserver_certrequest.crs -CA mycacert.pem -CAkey cakey.pem -CAcreateserial -out smserver_cert.pem
```

27. When OpenSSL prompts you, type the password for your certificate authority's private key. For example, `CAKeyPassword`.

OpenSSL stores the new signed certificate (`smserver_cert.pem`) in the `newcerts` directory.

**Tip:** To view the contents of the signed certificate, you can type following command:

```
openssl x509 -in smserver_cert.pem -text -noout
```

28. Copy the signed client certificate (`smserver_cert.pem`) to the OpenSSL server's Java platform bin folder.
29. Open the operating system's command prompt.
30. Change directories to the Java platform's bin folder.
31. Type the following command to import the Service Manager server's signed certificate into the server keystore.  
  

```
keytool -import -trustcacerts -alias myserver -keystore ./servercert.keystore -file smserver_cert.pem
```
32. When keytool prompts you to trust the private certificate authority's certificate, type `y`.



33. When keytool prompts you, type the password to your server's keystore file. For example,  
SMServerKeyPassword.

## Example: Viewing the contents of a cacerts file

To view the entries in a cacerts file, you can use the keytool utility provided with Sun J2SDK versions 1.4 or later. The following example uses the -list command to display the CA certificates in the cacerts file.

```
C:\j2sdk1.4.2_04\jre\bin>keytool -list -keystore ./cacerts
Enter keystore password:  changeit
```

```
Keystore type: jks
Keystore provider: SUN
```

Your keystore contains 15 entries

```
thawtepersonalfreemailca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): 1E:74:C3:86:3C:0C:35:C5:3E:C2:7F:EF:3C:AA:3C:D9
thawtepersonalbasicca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): E6:0B:D2:C9:CA:2D:88:DB:1A:71:0E:4B:78:EB:02:41
baltimorecodesigningca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): 90:F5:28:49:56:D1:5D:2C:B0:53:D4:4B:EF:6F:90:22
verisignclass3ca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): 78:2A:02:DF:DB:2E:14:D5:A7:5F:0A:DF:B6:8E:9C:5D
gtecybertrustglobalca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): CA:3D:D3:68:F1:03:5C:D0:32:FA:B8:2B:59:E8:5A:DB
thawtepersonalpremiumca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): 3A:B2:DE:22:9A:20:93:49:F9:ED:C8:D2:8A:E7:68:0D
thawteserverca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): C5:70:C4:A2:ED:53:78:0C:C8:10:53:81:64:CB:D0:1D
verisignclass4ca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): 1B:D1:AD:17:8B:7F:22:13:24:F5:26:E2:5D:4E:B9:10
baltimorecybertrustca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): AC:B6:94:A5:9C:17:E0:D7:91:52:9B:B1:97:06:A6:E4
verisignclass1ca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): 51:86:E8:1F:BC:B1:C3:71:B5:18:10:DB:5F:DC:F6:20
verisignserverca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): 74:7B:82:03:43:F0:00:9E:6B:B3:EC:47:BF:85:A5:93
thawtepremiumserverca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): 06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A
gtecybertrustca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): C4:D7:F0:B2:A3:C5:7D:61:67:F0:04:CD:43:D3:BA:58
gtecybertrust5ca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): 7D:6C:86:E4:FC:4D:D1:0B:00:BA:22:BB:4E:7C:6A:8E
verisignclass2ca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): EC:40:7D:2B:76:52:67:05:2C:EA:F2:3A:4F:65:F0:D8
```

## Update the cacerts keystore file

### **Applies to User Roles:**

System Administrator

If you use a private certificate authority to generate an SSL server certificate, you can add the private certificate authority to the list of trusted certificate authorities that exist in the Java cacerts keystore file. Sun distributes this file with JSSE and with JDK version 1.4.x and later releases. You can then distribute this updated cacerts file to your HP Service Manager clients so that they can validate the server's signed certificate.

**Note:** This procedure requires that you install a Java SDK of version 1.4.x or later on the server where you installed your private certificate authority.

To update the cacerts keystore file:

1. Log on to server where you installed your private certificate authority.
2. Open the operating systems command prompt.
3. Change directories to the Java SDK bin folder.
4. Type the following command to import your private certificate authority's certificate (for example, cacert.pem) into the Java cacerts file that you publish to the rest of your network. Change the path and variables as necessary.  

```
keytool -import -keystore ./cacerts -trustcacerts -file cacert.pem -storepass changeit
```
5. When keytool prompts you, type y to trust the private certificate authority's certificate.

## Requirements for required SSL encryption and client authentication

This configuration is intended for customers who:

- Want to require SSL encryption for all connections
- Want to protect against complex SSL-related attacks

- Want to authenticate that the HP Service Manager server is a valid host
- Want to authenticate that the HP Service Manager clients are valid hosts

### **Certificates required**

You must create or obtain the following certificates for SSL encryption.

- Certificate authority certificate \*
- Keystore containing the certificate authority certificate \*
- HP Service Manager server certificate
- Web tier certificate
- Windows client certificates

\* A typical SSL configuration uses a single certificate authority to issue certificates for all authenticated components. If, however, you use multiple certificate authorities to sign your certificates, then you need to obtain a certificate for each certificate authority.

### **Private keys required**

You must create or obtain the following private keys for SSL encryption.

- Certificate authority's private key \*
- HP Service Manager server's private key

\* This key is only necessary if you are managing your own private certificate authority.

### **Parameters required in the server configuration file (sm.ini)**

- keystoreFile – identify the keystore file containing the HP Service Manager server's certificate and private key
- keystorePass – identify the password to the keystore file containing the HP Service Manager server's certificate and private key
- ssl:1
- ssl\_reqClientAuth:1
- sslConnector:1
- truststoreFile – identify the keystore file containing the certificate authority's certificate

- **truststorePass** – identify the password to the keystore file containing the certificate authority's certificate

#### **Parameters required in the web tier configuration file (web.xml)**

You must set the following web parameters.

- **cacerts** – identify the keystore file containing the certificate authority that signed the server's certificate
- **keystore** – identify the keystore containing the web tier's client certificate
- **customize-folder** – specify the absolute path to a folder on the web tier host in which the `webtier.properties` file is located

#### **Parameter required in the <Customize-Folder>/config/webtier.properties file**

You must set the following web parameter:

**keystorePassword** – identify the password for the web tier's keystore

**Note:** The **keystorePassword** parameter has been removed from the web tier configuration file (`web.xml`) since Service Manager 9.34p2. You must enter your web client keystore password in a `webtier.properties` file. For details, see ["Encryption of client keystore passwords" on page 13](#).

#### **Windows client preferences required**

You must set the following preferences from the **Window > Preferences > HP Service Manager > Security** menu.

- **CA certificates file** – identify the keystore file containing the certificate authority that signed the server's certificate
- **Keystore file** – identify the keystore containing the Windows client's certificate
- **Keystore password** – identify the password for the Windows client's keystore

#### **Other requirements**

You must do the following additional steps to ensure that HP Service Manager can use your private certificates.

- Add your private certificate authority's certificate to a keystore that your Web and Windows clients can access
- Ensure that the HP Service Manager client's host name matches the common name (CN) listed in the client's signed certificate

- Ensure that the HP Service Manager server's host name matches the common name (CN) listed in the server's signed certificate

## Example: Enabling required SSL encryption and client authentication

The following example describes the following SSL configuration.

- Requiring SSL encryption using the HP Service Manager server's signed certificate
- Requiring client authentication using the HP Service Manager client's signed certificate

**Note:** This example builds on information presented in the topics ["Example: Generating a server certificate with OpenSSL" on page 93](#) and ["Example: Generating a client certificate with OpenSSL" on page 97](#).

1. Generate a signed server certificate for the HP Service Manager server.
2. Generate a signed client certificate for each HP Service Manager client.
3. Install the signed certificates and supporting keys on the HP Service Manager server.  
Copy the following keystore files into the RUN folder of the HP Service Manager server.
  - servercert.keystore – This keystore file contains the HP Service Manager server's signed certificate and private key
  - cacerts – This keystore file contains the certificate and private key of the certificate authority that signed the server's certificate
4. Install the certificate authority's certificate on your HP Service Manager clients.  
Copy the cacerts keystore containing your private certificate authority's certificate to the default certificate paths of your clients.
  - *<Web application server installation path>\WEB-INF* folder of the HP Service Manager web tier
  - *<Windows client installation path>\plugins\com.hp.ov.sm.client.common\_x.xx* folder of your HP Service Manager Windows clients
5. Stop the HP Service Manager server.
6. Open the HP Service Manager initialization file (sm.ini) with a text editor.

7. Add the following parameters to require SSL encryption using the HP Service Manager server's signed certificate.
  - `keystoreFile:servercert.keystore` – identifies the keystore file containing the HP Service Manager server's certificate and private key
  - `keystorePass:changeit` – identifies the password to the keystore file containing the HP Service Manager server's certificate and private key
  - `truststoreFile:cacerts` – identifies the keystore file containing the certificate authority's certificate
  - `truststorePass:changeit` – identifies the password to the keystore file containing the certificate authority's certificate
  - `ssl:1` – Requires SSL encryption using the server's signed certificate.
  - `ssl_reqClientAuth:1` – Requires HP Service Manager clients to present signed certificates to connect to the server.
  - `sslConnector:1` – requires Service Manager clients to use an HTTPS port when communicating with the server.
8. Save the HP Service Manager initialization file.
9. Restart the HP Service Manager server.
10. Configure HP Service Manager web clients to validate the HP Service Manager server's signed certificate and present signed client certificates. To do so, stop the web application server running the Web tier, open the web configuration file (`web.xml`) in a text editor, and perform the following steps:
  - a. Set **cacerts** to the keystore containing your server's certificate authority, for example `cacerts`. You copied this keystore to the `WEB-INF` folder.
  - b. Set **keystore** to the keystore containing your web tier's signed certificate, for example `clientcerts`. You created this keystore when you created the client certificate request.
  - c. Set **customize-folder** to a folder on the web tier host.
  - d. Create an empty `webtier.properties` file in the `<Customize-Folder>/config` directory. Later, you will specify the keystore password in the `webtier.properties` file.

**Note:** The **keystorePassword** parameter has been removed from the web tier configuration file (`web.xml`) since Service Manager 9.34p2. You must enter your web client keystore password in a `webtier.properties` file located in the `<Customize-Folder>/config` directory.

- e. Save the `web.xml` file.
  - f. In the `webtier.properties` file, set the **keystorePassword** parameter to the password to access the client keystore. For details, see ["Encryption of client keystore passwords" on page 13](#).
11. Configure HP Service Manager Windows clients to validate the HP Service Manager server's signed certificate and present signed client certificates.
- Click **Window > Preferences > HP Service Manager > Security**, and do the following:
- Set **CA Certificates File** to the `cacerts` keystore you copied to the `<Windows client installation path>\plugins\com.hp.ov.sm.client.common_x.xx` folder.
  - Set **Keystore File** to the keystore containing your Windows client's signed certificate, for example `clientcerts`. You created this keystore when you created the client certificate request.
  - Set **Keystore password** to the password required to access your client keystore. For example, `ClientKeyPassword`. You created this keystore password when you created the client certificate request.

## Example: Generating a client certificate with OpenSSL

The following example describes how to create a signed client certificate using the OpenSSL toolkit as a private certificate authority. This example also uses the `keytool` utility available with the Sun Microsystems™ standard Java Development Kit. You can use a client certificate to validate that the client is authorized to connect to HP Service Manager server or as part of a trusted sign-on configuration.

**Note:** This example builds on information presented in ["Example: Generating a server certificate with OpenSSL" on page 93](#). The information contained in this example regarding OpenSSL technology is provided by HP as a courtesy to our customers and partners. This documentation does not replace an OpenSSL reference, and HP encourages you to conduct additional research regarding OpenSSL technology by consulting with sources outside of this document. HP hereby disclaims all liability associated with the use and accuracy of this information. As OpenSSL technology evolves, HP may or may not update this reference.

1. Log on to server where you installed your OpenSSL private certificate authority, and open the operating system's command prompt.
2. Change directories to the Java platform's bin folder.
3. Type the following command to create a private key and keystore for your Service Manager client. For example, to create a private key and keystore for your Service Manager web tier, type:  
`keytool -genkey -alias clients -keystore <clientcerts>.keystore`

**Note:** When you repeat this step for multiple clients, replace <clientcerts> (and also <client> in the following steps) with a name that can identify every single client. For example, you can use the FQDN for each Windows client, and use the FQDN or smwebtier for your web tier client.

4. When keytool prompts you, type the password phrase you want to use to protect your Service Manager client's keystore file. For example, ClientKeyPassword.
5. When keytool prompts you for your first and last name, type the fully qualified host name of your Service Manager client system.
6. When keytool prompts you for the organization unit, organization, city or locality, state or province, and two-letter country code, type the identification information for your company.
7. Verify the information you provided and type yes if it is correct.
8. When keytool prompts you for the password phrase to use for your Service Manager web tier's private key, press ENTER to use the same password as you created for the keystore.

**Note:** The password for the private key must match the password for the keystore file.

9. Type the following command to create a certificate request for your Service Manager client. For example, to create a certificate request for your Service Manager web tier, type:  
`keytool -certreq -alias clients -keystore <clientcerts>.keystore -file <client>_certrequest.crs`
10. When keytool prompts you, type the password for the Service Manager client's keystore file (from step 4). For example, ClientKeyPassword.
11. Copy the Service Manager client's certificate request (For example, <client>\_certrequest.crs) to the OpenSSL bin folder.



12. Change directories to the OpenSSL bin folder.
13. Type the following command to sign the Service Manager client's certificate request with your private certificate authority:  

```
openssl x509 -req -days 365 -in <client>_certrequest.crs -CA mycacert.pem -CAkey cakey.pem -CAcreateserial -out <client>_cert.pem
```
14. When OpenSSL prompts you, type the password for your certificate authority's private key. For example, CAKeyPassword.

OpenSSL stores the new signed certificate (<client>\_cert.pem) in the newcerts directory.

**Tip:** To view the contents of the signed certificate, you can type following command:

```
openssl x509 -in <client>_cert.pem -text -noout
```

15. Copy the signed client certificate (<client>\_cert.pem) to the OpenSSL server's Java platform bin folder.
16. Open the operating system's command prompt.
17. Change directories to the Java platform's bin folder.
18. Type the following command to import the Service Manager client's signed certificate into a client keystore.  

```
keytool -import -trustcacerts -alias clients -keystore ./<clientcerts>.keystore -file <client>_cert.pem
```
19. When keytool prompts you to trust the private certificate authority's certificate, type y.
20. Copy the updated client keystore (<clientcerts>.keystore) to the default certificate path of your client:
  - WEB-INF folder of the Service Manager Web tier
  - <Windows client installation path>\plugins\com.hp.ov.sm.client.common\_x.xx folder of your Service Manager Windows clients
21. If you are using a trusted clients or trusted sign-on implementation, do the following:
  - a. Import each client certificate you want to be part of the list of trusted clients to a trusted clients keystore. To do so, type the following command:

```
keytool -import -alias client1 -file <client>_cert.pem -keystore  
trustedclients.keystore
```

- b. Copy the trusted clients keystore (trustedclients.keystore) to the Service Manager server's RUN folder.

## Example: Generating a server certificate with OpenSSL

The following example describes how to create a signed server certificate using the OpenSSL toolkit as a private certificate authority. This example also uses the keytool utility available with the Sun Microsystems™ standard Java Development Kit. You can use a server certificate to enable SSL encryption from the HP Service Manager server.

**Note:** The information contained in this example regarding OpenSSL technology is provided by HP as a courtesy to our customers and partners. This documentation does not replace an OpenSSL reference, and HP encourages you to conduct additional research regarding OpenSSL technology by consulting with sources outside of this document. HP hereby disclaims all liability associated with the use and accuracy of this information. As OpenSSL technology evolves, HP may or may not update this reference.

1. Obtain and install a Java platform on the server you want to use as your private certificate authority. See the Sun Microsystems™ Java Technology site for the necessary software and documentation.
2. Obtain and install OpenSSL on the server you want to use as your private certificate authority. See the OpenSSL Web site for the necessary software and documentation.

**Caution:** HP strongly recommends that you do not install your private certificate authority on the same server as your Service Manager production server.

3. Create and configure an openssl.conf file in the bin folder of your OpenSSL installation.
4. Open the operating system's command prompt on the private certificate authority server.
5. Change directories to the OpenSSL bin folder.
6. Type the following command to create the private key for your private certificate authority:  

```
openssl genrsa -des3 -out cakey.pem 2048
```

7. When OpenSSL prompts you, type the password phrase you want to use to protect your certificate authority's private key file (`cakey.pem`). For example, `CAKeyPassword`.  
You must use the same password phrase each time you sign a certificate request with your private certificate authority. If you forget this password, you must repeat the steps to create another certificate authority private key.
8. Type the following command to create a public certificate for your private certificate authority:  

```
openssl req -new -key cakey.pem -x509 -days 1095 -out mycacert.pem -config  
.\openssl.conf
```
9. Change directories to the Java platform's `bin` folder.
10. Type the following command to import your private certificate authority's certificate into the Java `cacerts` file that you will publish to the rest of your network.  

```
keytool -import -keystore ./cacerts -trustcacerts -file mycacert.pem -storepass  
changeit
```
11. When `keytool` prompts you, type `y` to trust the private certificate authority's certificate.
12. Install the updated Java `cacerts` file on the Service Manager server. Copy the `cacerts` file to the `RUN` folder of the Service Manager server. You can also copy the `cacerts` file to the `JRE\lib\security` folder of your Java Run Time Environment (RTE) or Java Development Kit (JDK).
13. Install the updated Java `cacerts` file on the Service Manager web tier. Copy the `cacerts` file to the `<web application server installation path>\WEB-INF` folder of the Service Manager web tier.
14. Install the updated Java `cacerts` file on the Service Manager Windows client. Copy the `cacerts` file to the `<Windows client installation path>\plugins\com.hp.ov.sm.client.common_x.xx` folder of the Service Manager Windows client.
15. Change directories to the Java platform's `bin` folder.
16. Type the following command to create a private key and keystore for your Service Manager server.  

```
keytool -genkey -alias myserver -keystore servercert.keystore
```
17. When `keytool` prompts you, type the password phrase you want to use to protect your Service Manager server's keystore file. For example, `SMServerKeyPassword`.

**Note:** Do not use the same password as your private certificate authority key password.

18. When keytool prompts you for your first and last name, type the fully qualified host name of your Service Manager server system.

**Note:** This is the same name your clients will use to connect to the Service Manager server.

19. When keytool prompts you for the organization unit, organization, city or locality, state or province, and two-letter country code, type the identification information for your company.
20. Verify the information you provided and type yes if it is correct.
21. When keytool prompts you for the password phrase to use for your Service Manager server's private key, press ENTER to use the same password as you created for the keystore.

**Note:** The password for the private key must match the password for the keystore file.

22. Type the following command to create a certificate request for your Service Manager server. For example, to create a certificate request for your HP Service Manager server type:

```
keytool -certreq -alias myserver -keystore servercert.keystore -file smserver_
certrequest.crs
```

23. When keytool prompts you, type the password for the HP Service Manager server's keystore file (from step 17). For example, SMServerKeyPassword.
24. Copy the Service Manager server's certificate request (For example, smserver\_certrequest.crs) to the OpenSSL bin folder.
25. Change directories to the OpenSSL bin folder.
26. Type the following command to sign the Service Manager server's certificate request with your private certificate authority:  

```
openssl x509 -req -days 365 -in smserver_certrequest.crs -CA mycacert.pem -
CAkey cakey.pem -CAcreateserial -out smserver_cert.pem
```
27. When OpenSSL prompts you, type the password for your certificate authority's private key. For example, CAKeyPassword.

OpenSSL stores the new signed certificate (smserver\_cert.pem) in the newcerts directory.

**Tip:** To view the contents of the signed certificate, you can type following command:

```
openssl x509 -in smsserver_cert.pem -text -noout
```

28. Copy the signed client certificate (smsserver\_cert.pem) to the OpenSSL server's Java platform bin folder.
29. Open the operating system's command prompt.
30. Change directories to the Java platform's bin folder.
31. Type the following command to import the Service Manager server's signed certificate into the server keystore.  

```
keytool -import -trustcacerts -alias myserver -keystore ./servercert.keystore -file smsserver_cert.pem
```
32. When keytool prompts you to trust the private certificate authority's certificate, type y.
33. When keytool prompts you, type the password to your server's keystore file. For example, SMServerKeyPassword.

## Example: Viewing the contents of a cacerts file

To view the entries in a cacerts file, you can use the keytool utility provided with Sun J2SDK versions 1.4 or later. The following example uses the -list command to display the CA certificates in the cacerts file.

```
C:\j2sdk1.4.2_04\jre\bin>keytool -list -keystore ./cacerts
Enter keystore password: changeit
```

```
Keystore type: jks
Keystore provider: SUN
```

Your keystore contains 15 entries

```
thawtepersonalfreemailca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): 1E:74:C3:86:3C:0C:35:C5:3E:C2:7F:EF:3C:AA:3C:D9
thawtepersonalbasicca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): E6:0B:D2:C9:CA:2D:88:DB:1A:71:0E:4B:78:EB:02:41
baltimorecodesigningca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): 90:F5:28:49:56:D1:5D:2C:B0:53:D4:4B:EF:6F:90:22
verisignclass3ca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): 78:2A:02:DF:DB:2E:14:D5:A7:5F:0A:DF:B6:8E:9C:5D
gtecybertrustglobalca, May 10, 2002, trustedCertEntry,
```

Certificate fingerprint (MD5): CA:3D:D3:68:F1:03:5C:D0:32:FA:B8:2B:59:E8:5A:DB  
 thawtepersonalpremiumca, Feb 12, 1999, trustedCertEntry,  
 Certificate fingerprint (MD5): 3A:B2:DE:22:9A:20:93:49:F9:ED:C8:D2:8A:E7:68:0D  
 thawteserverca, Feb 12, 1999, trustedCertEntry,  
 Certificate fingerprint (MD5): C5:70:C4:A2:ED:53:78:0C:C8:10:53:81:64:CB:D0:1D  
 verisignclass4ca, Jun 29, 1998, trustedCertEntry,  
 Certificate fingerprint (MD5): 1B:D1:AD:17:8B:7F:22:13:24:F5:26:E2:5D:4E:B9:10  
 baltimorecybertrustca, May 10, 2002, trustedCertEntry,  
 Certificate fingerprint (MD5): AC:B6:94:A5:9C:17:E0:D7:91:52:9B:B1:97:06:A6:E4  
 verisignclass1ca, Jun 29, 1998, trustedCertEntry,  
 Certificate fingerprint (MD5): 51:86:E8:1F:BC:B1:C3:71:B5:18:10:DB:5F:DC:F6:20  
 verisignserverca, Jun 29, 1998, trustedCertEntry,  
 Certificate fingerprint (MD5): 74:7B:82:03:43:F0:00:9E:6B:B3:EC:47:BF:85:A5:93  
 thawtepremiumserverca, Feb 12, 1999, trustedCertEntry,  
 Certificate fingerprint (MD5): 06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A  
 gtecybertrustca, May 10, 2002, trustedCertEntry,  
 Certificate fingerprint (MD5): C4:D7:F0:B2:A3:C5:7D:61:67:F0:04:CD:43:D3:BA:58  
 gtecybertrust5ca, May 10, 2002, trustedCertEntry,  
 Certificate fingerprint (MD5): 7D:6C:86:E4:FC:4D:D1:0B:00:BA:22:BB:4E:7C:6A:8E  
 verisignclass2ca, Jun 29, 1998, trustedCertEntry,  
 Certificate fingerprint (MD5): EC:40:7D:2B:76:52:67:05:2C:EA:F2:3A:4F:65:F0:D8

## Add a client certificate to the web tier

You can add a client certificate to your web tier to support client host validation or trusted sign-on.

**Note:** The following procedure assumes that you have already generated or obtained a client certificate for your web tier and imported the client certificate into a keystore.

1. Log on to the web tier system.
2. Stop the web tier web application server.
3. Copy the keystore containing the client certificate to one of two locations:
  - The web tier's web application WEB-INF folder
  - A network share accessible to your web tier
4. Open the web configuration file (`web.xml`) in a text editor, and perform the following steps:
  - a. Set the **keystore** parameter to the path of the keystore containing the web tier's certificate.
  - b. Set **customize-folder** to a folder on the web tier host.
  - c. Create an empty `webtier.properties` file in the `<Customize-Folder>/config` directory. You will

specify the keystore password in the webtier.properties file later.

**Note:** The **keystorePassword** parameter has been removed from the web tier configuration file (web.xml) since Service Manager 9.34p2. You must enter your web client keystore password in a webtier.properties file located in the <Customize-Folder>/config directory.

- d. Save the web.xml file.
5. In the webtier.properties file, set the **keystorePassword** parameter to the password to access the client keystore. For details, see ["Encryption of client keystore passwords" on page 13](#).
6. Restart your web tier web application server.

## Add a client certificate to the Windows client

You can add a client certificate to your Windows client to support client host validation or trusted sign-on.

**Note:** The following procedure assumes that you have already generated or obtained a client certificate for your Windows client and imported the client certificate into a keystore.

1. Log on to the Windows client system.
2. Copy the keystore containing the client certificate to one of two locations:
  - `<Windows client installation path>\plugins\com.hp.ov.sm.client.common_x.xx` folder
  - A network share accessible to your Windows clients
3. Open the Windows client.
4. Click **Window > Preferences > HP Service Manager > Security**.
5. Click **Browse** for the **CA Certificate File** field.
6. Browse to the path of the **cacerts** keystore containing the server's CA certificate.
7. Click **Browse** for the **Client Keystore File** field.
8. Browse to the path of the keystore containing the Windows client's certificate.

9. In the **Client Keystore Password** field, type the password to access the client keystore.
10. Click **OK** to accept the path.
11. Click **OK** to save your changes.

## Update the cacerts keystore file

### **Applies to User Roles:**

System Administrator

If you use a private certificate authority to generate an SSL server certificate, you can add the private certificate authority to the list of trusted certificate authorities that exist in the Java cacerts keystore file. Sun distributes this file with JSSE and with JDK version 1.4.x and later releases. You can then distribute this updated cacerts file to your HP Service Manager clients so that they can validate the server's signed certificate.

**Note:** This procedure requires that you install a Java SDK of version 1.4.x or later on the server where you installed your private certificate authority.

To update the cacerts keystore file:

1. Log on to server where you installed your private certificate authority.
2. Open the operating systems command prompt.
3. Change directories to the Java SDK bin folder.
4. Type the following command to import your private certificate authority's certificate (for example, cacert.pem) into the Java cacerts file that you publish to the rest of your network. Change the path and variables as necessary.  

```
keytool -import -keystore ./cacerts -trustcacerts -file cacert.pem -storepass changeit
```
5. When keytool prompts you, type y to trust the private certificate authority's certificate.

## Use keytool to create a certificate request

### **Applies to User Roles:**

System Administrator



You can use the keytool utility provided with the Sun Microsystems™ Java Development Kit to create a certificate request.

To use keytool to create a certificate request:

1. Open your operating system's command prompt.
2. Change directories to your JDK's bin folder.
3. Type the following command to create a client certificate request.

```
keytool -certreq -alias clients -keystore clientcerts -file smwebtier_  
certrequest.crs
```

You can define your own names for the -alias, -keystore, and -file parameters. The names above are examples.

4. When the keytool utility prompts you, type the Keystore password.  
Keytool exports the client certificate key to the file and path you specified with the -file parameter.

## Use keytool to create a private key

### **Applies to User Roles:**

System Administrator

You can use the keytool utility provided with the Sun Microsystems™ Java Development Kit to produce a private key in a keystore.

To use keytool to create a private key:

1. Open your operating system's command prompt.
2. Change directories to your JDK's bin folder.
3. Type the following command to create a client private key.

```
keytool -genkey -alias clients -keystore clientcerts
```

You can define your own names for the -alias and -keystore parameters. The names above are examples.

4. When the keytool utility prompts you, type the following information:
  - Keystore password
  - Fully qualified domain name of host

- Organizational unit
  - Name of organization
  - City or locality
  - State or province
  - Two letter country code
5. Review the contact information and type yes to accept it.
  6. Type in the password you want to use for the client key.

**Tip:** You can press ENTER to use the same password for the key as you typed for the keystore.

## Requirements for required SSL encryption and trusted clients

This configuration is intended for customers who:

- Want to require SSL encryption for all connections
- Want to protect against complex SSL-related attacks
- Want to authenticate that the HP Service Manager server is a valid host
- Want to authenticate that the HP Service Manager clients are valid hosts
- Want to restrict access to the server to only those clients whose certificates are in a list of trusted clients

### **Certificates required**

You must create or obtain the following certificates for SSL encryption.

- Certificate authority certificate \*
- Certificate list containing the certificates of web and Windows clients that are allowed to connect to the server
- Keystore containing the certificate authority certificate \*

- HP Service Manager server certificate
- Web tier certificate
- Windows client certificates

\* A typical SSL configuration uses a single certificate authority to issue certificates for all authenticated components. If, however, you use multiple certificate authorities to sign your certificates, then you need to obtain a certificate for each certificate authority.

#### **Private keys required**

You must create or obtain the following private keys for SSL encryption.

- Certificate authority's private key \*
- HP Service Manager server's private key

\* This key is only necessary if you are managing your own private certificate authority.

#### **Parameters required in the server configuration file (sm.ini)**

- keystoreFile – identify the keystore file containing the HP Service Manager server's certificate and private key
- keystorePass – identify the password to the keystore file containing the HP Service Manager server's certificate and private key
- ssl:1
- ssl\_reqClientAuth:2
- ssl\_trustedClientsJKS – identify the keystore file containing the list of trusted client certificates allowed to connect to the server
- ssl\_trustedClientsPwd – identify the password to the keystore file containing the list of trusted client certificates
- sslConnector:1
- truststoreFile – identify the keystore file containing the certificate authority's certificate
- truststorePass – identify the password to the keystore file containing the certificate authority's certificate

#### **Parameters required in the web tier configuration file (web.xml)**

You must set the following web parameters.

- `cacerts` – identify the keystore file containing the certificate authority that signed the server's certificate
- `keystore` – identify the keystore containing the web tier's client certificate
- `customize-folder` – specify the absolute path to a folder on the web tier host in which the `webtier.properties` file is located

**Parameter required in the <Customize-Folder>/config/webtier.properties file**

You must set the following web parameter:

`keystorePassword` – identify the password for the web tier's keystore

**Note:** The `keystorePassword` parameter has been removed from the web tier configuration file (`web.xml`) since Service Manager 9.34p2. You must enter your web client keystore password in a `webtier.properties` file. For details, see ["Encryption of client keystore passwords" on page 13](#).

**Windows client preferences required**

You must set the following preferences from the **Window > Preferences > HP Service Manager > Security** menu.

- **CA certificates file** – identify the keystore file containing the certificate authority that signed the server's certificate
- **Keystore file** – identify the keystore containing the Windows client's certificate
- **Keystore password** – identify the password for the Windows client's keystore

**Other requirements**

You must do the following additional steps to ensure that HP Service Manager can use your private certificates.

- Add your private certificate authority's certificate to a keystore that your Web and Windows clients can access
- Ensure that the HP Service Manager client's host name matches the common name (CN) listed in the client's signed certificate
- Ensure that the HP Service Manager server's host name matches the common name (CN) listed in the server's signed certificate

## Example: Enabling required SSL encryption and trusted clients

The following example describes the following SSL configuration.

- Requiring SSL encryption using the HP Service Manager server's signed certificate
- Requiring client authentication using the Service Manager client's signed certificate
- Requiring trusted client authentication using a list of trusted client certificates

**Note:** This example builds on information presented in the generating a server certificate and generating a client certificate examples.

1. Generate a signed server certificate for the Service Manager server. See ["Example: Generating a server certificate with OpenSSL" on page 93](#).
2. Generate a signed client certificate for each Service Manager client. See ["Example: Generating a client certificate with OpenSSL" on page 97](#).

**Note:** This step involves importing each Service Manager client's signed certificate into a trusted clients keystore (trustedclients.keystore) by using the keytool import command. To enable trusted sign-on, you must do so for each client certificate you want to be part of the list of trusted clients.

3. Install the signed certificates and supporting keys on the Service Manager server. To do so, copy the following keystore files into the RUN folder of the Service Manager server.
  - servercert.keystore – This keystore file contains the Service Manager server's signed certificate and private key
  - cacerts – This keystore file contains the certificate and private key of the certificate authority that signed the server's certificate
  - trustedclients.keystore – This keystore contains the signed certificates of your Service Manager server's trusted clients
4. Stop the Service Manager server.
5. Open the Service Manager initialization file (sm.ini) with a text editor.

6. Add the following parameters to require SSL encryption using the Service Manager server's signed certificate.
  - keystoreFile:servercert.keystore – identifies the keystore file containing the Service Manager server's certificate and private key
  - keystorePass:changeit – identifies the password to the keystore file containing the Service Manager server's certificate and private key
  - ssl:1 – Requires SSL encryption using the server's signed certificate.
  - ssl\_reqClientAuth:2 – Requires Service Manager clients to present signed certificates to connect to the server and also be on the list of trusted clients.
  - ssl\_trustedClientsJKS:trustedclients.keystore – identifies the keystore containing the signed certificates of trusted Service Manager clients
  - ssl\_trustedClientsPwd:ClientKeyPassword – identifies the password to the keystore file containing the signed certificates of trusted Service Manager clients
  - sslConnector:1 – requires Service Manager clients to use an HTTPS port when communicating with the server.
  - truststoreFile:cacerts – identifies the keystore file containing the certificate authority's certificate
  - truststorePass:changeit – identifies the password to the keystore file containing the certificate authority's certificate
7. Save the Service Manager initialization file.
8. Restart the Service Manager server.
9. Copy the following keystore files to the web tier's WEB-INF folder:
  - cacerts – This keystore file contains the certificate and private key of the certificate authority that signed the server's certificate
  - <clientcerts>.keystore – This keystore contains the signed certificate of your Service Manager web tier client
10. Copy the following keystore files to each Windows client's *<Windows client installation path>\plugins\com.hp.ov.sm.client.common\_x.xx* folder.

- `cacerts` – This keystore file contains the certificate and private key of the certificate authority that signed the server's certificate
- `<clientcerts>.keystore` – This keystore contains the signed certificate of each Windows client

**Note:** You should have created a separate `<clientcerts>.keystore` for each Windows client.

11. Configure Service Manager web clients to validate the Service Manager server's signed certificate and present signed client certificates.
  - a. Stop the web application server running the web tier, open the web configuration file (`web.xml`) in a text editor.
  - b. Set `cacerts` to the keystore containing your server's certificate authority, for example `/WEB-INF/cacerts`. You copied this keystore from the `JRE\lib\security` folder of your Java Run Time Environment (RTE) or Java Development Kit (JDK) to the web tier's `WEB-INF` folder.
  - c. Set `keystore` to the keystore containing your web tier's signed certificate, for example `/WEB-INF/<clientcerts>.keystore`. You created this keystore when you created the client certificate request.
  - d. Set `customize-folder` to a folder on the web tier host in which your `webtier.properties` file is located. You created a `webtier.properties` file in the `<Customize-Folder>/config` directory when you set the keystore password.
  - e. Set `keystorePassword` to the password required to access your web tier client keystore, for example `ClientKeyPassword`. You created this keystore password when you created the web tier client certificate request.

**Note:** The **keystorePassword** parameter has been removed from the web tier configuration file `web.xml` since Service Manager 9.34p2, and you should enter your web client keystore password in a `webtier.properties` file. For more information about how to set the **keystorePassword** parameter, see ["Encryption of client keystore passwords" on page 13](#).

- f. Set `ssl` to `true`.
  - g. Set `serverHost` to the fully-qualified domain name of the Service Manager server. For example: `myserver.mydomain.com`.

12. Configure Service Manager Windows clients to validate the Service Manager server's signed certificate and present signed client certificates. Do the following on each Windows client.
  - a. Click **Window > Preferences > Service Manager > Security**.
  - b. Set **CA Certificates File** to the cacerts keystore you copied to the `<windows client installation path>\plugins\com.hp.ov.sm.client.common_x.xx` folder.
  - c. Set **Keystore File** to the keystore containing your Windows client's signed certificate, for example `<clientcerts>.keystore`. You created this keystore when you created the Windows client certificate request.
  - d. Set **Keystore password** to the password required to access the Windows client keystore. For example, **ClientKeyPassword**. You created this keystore password when you created the Windows client certificate request.
  - e. Update your Windows client connections by selecting **Use SSL Encryption** on their Advanced tab.

## Example: Generating a client certificate with OpenSSL

The following example describes how to create a signed client certificate using the OpenSSL toolkit as a private certificate authority. This example also uses the keytool utility available with the Sun Microsystems™ standard Java Development Kit. You can use a client certificate to validate that the client is authorized to connect to HP Service Manager server or as part of a trusted sign-on configuration.

**Note:** This example builds on information presented in ["Example: Generating a server certificate with OpenSSL" on page 93](#). The information contained in this example regarding OpenSSL technology is provided by HP as a courtesy to our customers and partners. This documentation does not replace an OpenSSL reference, and HP encourages you to conduct additional research regarding OpenSSL technology by consulting with sources outside of this document. HP hereby disclaims all liability associated with the use and accuracy of this information. As OpenSSL technology evolves, HP may or may not update this reference.

1. Log on to server where you installed your OpenSSL private certificate authority, and open the operating system's command prompt.
2. Change directories to the Java platform's bin folder.



3. Type the following command to create a private key and keystore for your Service Manager client. For example, to create a private key and keystore for your Service Manager web tier, type:

```
keytool -genkey -alias clients -keystore <clientcerts>.keystore
```

**Note:** When you repeat this step for multiple clients, replace <clientcerts> (and also <client> in the following steps) with a name that can identify every single client. For example, you can use the FQDN for each Windows client, and use the FQDN or smwebtier for your web tier client.

4. When keytool prompts you, type the password phrase you want to use to protect your Service Manager client's keystore file. For example, ClientKeyPassword.
5. When keytool prompts you for your first and last name, type the fully qualified host name of your Service Manager client system.
6. When keytool prompts you for the organization unit, organization, city or locality, state or province, and two-letter country code, type the identification information for your company.
7. Verify the information you provided and type yes if it is correct.
8. When keytool prompts you for the password phrase to use for your Service Manager web tier's private key, press ENTER to use the same password as you created for the keystore.

**Note:** The password for the private key must match the password for the keystore file.

9. Type the following command to create a certificate request for your Service Manager client. For example, to create a certificate request for your Service Manager web tier, type:  

```
keytool -certreq -alias clients -keystore <clientcerts>.keystore -file  
<client>_certrequest.crs
```
10. When keytool prompts you, type the password for the Service Manager client's keystore file (from step 4). For example, ClientKeyPassword.
11. Copy the Service Manager client's certificate request (For example, <client>\_certrequest.crs) to the OpenSSL bin folder.
12. Change directories to the OpenSSL bin folder.
13. Type the following command to sign the Service Manager client's certificate request with your private certificate authority:

```
openssl x509 -req -days 365 -in <client>_certrequest.crs -CA mycacert.pem -CAkey cakey.pem -CAcreateserial -out <client>_cert.pem
```

14. When OpenSSL prompts you, type the password for your certificate authority's private key. For example, CAKeyPassword.

OpenSSL stores the new signed certificate (<client>\_cert.pem) in the newcerts directory.

**Tip:** To view the contents of the signed certificate, you can type following command:

```
openssl x509 -in <client>_cert.pem -text -noout
```

15. Copy the signed client certificate (<client>\_cert.pem) to the OpenSSL server's Java platform bin folder.
16. Open the operating system's command prompt.
17. Change directories to the Java platform's bin folder.
18. Type the following command to import the Service Manager client's signed certificate into a client keystore.  

```
keytool -import -trustcacerts -alias clients -keystore ./<clientcerts>.keystore -file <client>_cert.pem
```
19. When keytool prompts you to trust the private certificate authority's certificate, type y.
20. Copy the updated client keystore (<clientcerts>.keystore) to the default certificate path of your client:
  - WEB-INF folder of the Service Manager Web tier
  - <Windows client installation path>\plugins\com.hp.ov.sm.client.common\_x.xx folder of your Service Manager Windows clients
21. If you are using a trusted clients or trusted sign-on implementation, do the following:
  - a. Import each client certificate you want to be part of the list of trusted clients to a trusted clients keystore. To do so, type the following command:

```
keytool -import -alias client1 -file <client>_cert.pem -keystore trustedclients.keystore
```

- b. Copy the trusted clients keystore (trustedclients.keystore) to the Service Manager server's RUN folder.

## Example: Generating a server certificate with OpenSSL

The following example describes how to create a signed server certificate using the OpenSSL toolkit as a private certificate authority. This example also uses the keytool utility available with the Sun Microsystems™ standard Java Development Kit. You can use a server certificate to enable SSL encryption from the HP Service Manager server.

**Note:** The information contained in this example regarding OpenSSL technology is provided by HP as a courtesy to our customers and partners. This documentation does not replace an OpenSSL reference, and HP encourages you to conduct additional research regarding OpenSSL technology by consulting with sources outside of this document. HP hereby disclaims all liability associated with the use and accuracy of this information. As OpenSSL technology evolves, HP may or may not update this reference.

1. Obtain and install a Java platform on the server you want to use as your private certificate authority. See the Sun Microsystems™ Java Technology site for the necessary software and documentation.
2. Obtain and install OpenSSL on the server you want to use as your private certificate authority. See the OpenSSL Web site for the necessary software and documentation.

**Caution:** HP strongly recommends that you do not install your private certificate authority on the same server as your Service Manager production server.

3. Create and configure an openssl.conf file in the bin folder of your OpenSSL installation.
4. Open the operating system's command prompt on the private certificate authority server.
5. Change directories to the OpenSSL bin folder.
6. Type the following command to create the private key for your private certificate authority:  
`openssl genrsa -des3 -out cakey.pem 2048`
7. When OpenSSL prompts you, type the password phrase you want to use to protect your certificate authority's private key file (cakey.pem). For example, CAKeyPassword.  
You must use the same password phrase each time you sign a certificate request with your

private certificate authority. If you forget this password, you must repeat the steps to create another certificate authority private key.

8. Type the following command to create a public certificate for your private certificate authority:  
`openssl req -new -key cakey.pem -x509 -days 1095 -out mycacert.pem -config .\openssl.conf`
9. Change directories to the Java platform's bin folder.
10. Type the following command to import your private certificate authority's certificate into the Java cacerts file that you will publish to the rest of your network.  
`keytool -import -keystore ./cacerts -trustcacerts -file mycacert.pem -storepass changeit`
11. When keytool prompts you, type *y* to trust the private certificate authority's certificate.
12. Install the updated Java cacerts file on the Service Manager server. Copy the cacerts file to the RUN folder of the Service Manager server. You can also copy the cacerts file to the JRE\lib\security folder of your Java Run Time Environment (RTE) or Java Development Kit (JDK).
13. Install the updated Java cacerts file on the Service Manager web tier. Copy the cacerts file to the *<web application server installation path>\WEB-INF* folder of the Service Manager web tier.
14. Install the updated Java cacerts file on the Service Manager Windows client. Copy the cacerts file to the *<Windows client installation path>\plugins\com.hp.ov.sm.client.common\_x.xx* folder of the Service Manager Windows client.
15. Change directories to the Java platform's bin folder.
16. Type the following command to create a private key and keystore for your Service Manager server.  
`keytool -genkey -alias myserver -keystore servercert.keystore`
17. When keytool prompts you, type the password phrase you want to use to protect your Service Manager server's keystore file. For example, *SMServerKeyPassword*.

**Note:** Do not use the same password as your private certificate authority key password.

18. When keytool prompts you for your first and last name, type the fully qualified host name of your

Service Manager server system.

**Note:** This is the same name your clients will use to connect to the Service Manager server.

19. When keytool prompts you for the organization unit, organization, city or locality, state or province, and two-letter country code, type the identification information for your company.
20. Verify the information you provided and type **yes** if it is correct.
21. When keytool prompts you for the password phrase to use for your Service Manager server's private key, press **ENTER** to use the same password as you created for the keystore.

**Note:** The password for the private key must match the password for the keystore file.

22. Type the following command to create a certificate request for your Service Manager server. For example, to create a certificate request for your HP Service Manager server type:  

```
keytool -certreq -alias myserver -keystore servercert.keystore -file smserver_
certrequest.crs
```
23. When keytool prompts you, type the password for the HP Service Manager server's keystore file (from step 17). For example, `SMServerKeyPassword`.
24. Copy the Service Manager server's certificate request (For example, `smserver_certrequest.crs`) to the OpenSSL bin folder.
25. Change directories to the OpenSSL bin folder.
26. Type the following command to sign the Service Manager server's certificate request with your private certificate authority:  

```
openssl x509 -req -days 365 -in smserver_certrequest.crs -CA mycacert.pem -
CAkey cakey.pem -CAcreateserial -out smserver_cert.pem
```
27. When OpenSSL prompts you, type the password for your certificate authority's private key. For example, `CAKeyPassword`.

OpenSSL stores the new signed certificate (`smserver_cert.pem`) in the `newcerts` directory.

**Tip:** To view the contents of the signed certificate, you can type following command:

```
openssl x509 -in smserver_cert.pem -text -noout
```

28. Copy the signed client certificate (smserver\_cert.pem) to the OpenSSL server's Java platform bin folder.
29. Open the operating system's command prompt.
30. Change directories to the Java platform's bin folder.
31. Type the following command to import the Service Manager server's signed certificate into the server keystore.  

```
keytool -import -trustcacerts -alias myserver -keystore ./servercert.keystore -file smserver_cert.pem
```
32. When keytool prompts you to trust the private certificate authority's certificate, type y.
33. When keytool prompts you, type the password to your server's keystore file. For example, SMServerKeyPassword.

## Example: Viewing the contents of a cacerts file

To view the entries in a cacerts file, you can use the keytool utility provided with Sun J2SDK versions 1.4 or later. The following example uses the -list command to display the CA certificates in the cacerts file.

```
C:\j2sdk1.4.2_04\jre\bin>keytool -list -keystore ./cacerts
Enter keystore password: changeit
```

```
Keystore type: jks
Keystore provider: SUN
```

Your keystore contains 15 entries

```
thawtepersonalfreemailca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): 1E:74:C3:86:3C:0C:35:C5:3E:C2:7F:EF:3C:AA:3C:D9
thawtepersonalbasicca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): E6:0B:D2:C9:CA:2D:88:DB:1A:71:0E:4B:78:EB:02:41
baltimorecodesigningca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): 90:F5:28:49:56:D1:5D:2C:B0:53:D4:4B:EF:6F:90:22
verisignclass3ca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): 78:2A:02:DF:DB:2E:14:D5:A7:5F:0A:DF:B6:8E:9C:5D
gtecybertrustglobalca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): CA:3D:D3:68:F1:03:5C:D0:32:FA:B8:2B:59:E8:5A:DB
thawtepersonalpremiumca, Feb 12, 1999, trustedCertEntry,
```

Certificate fingerprint (MD5): 3A:B2:DE:22:9A:20:93:49:F9:ED:C8:D2:8A:E7:68:0D  
thawteserverca, Feb 12, 1999, trustedCertEntry,  
Certificate fingerprint (MD5): C5:70:C4:A2:ED:53:78:0C:C8:10:53:81:64:CB:D0:1D  
verisignclass4ca, Jun 29, 1998, trustedCertEntry,  
Certificate fingerprint (MD5): 1B:D1:AD:17:8B:7F:22:13:24:F5:26:E2:5D:4E:B9:10  
baltimorecybertrustca, May 10, 2002, trustedCertEntry,  
Certificate fingerprint (MD5): AC:B6:94:A5:9C:17:E0:D7:91:52:9B:B1:97:06:A6:E4  
verisignclass1ca, Jun 29, 1998, trustedCertEntry,  
Certificate fingerprint (MD5): 51:86:E8:1F:BC:B1:C3:71:B5:18:10:DB:5F:DC:F6:20  
verisignserverca, Jun 29, 1998, trustedCertEntry,  
Certificate fingerprint (MD5): 74:7B:82:03:43:F0:00:9E:6B:B3:EC:47:BF:85:A5:93  
thawtepremiumserverca, Feb 12, 1999, trustedCertEntry,  
Certificate fingerprint (MD5): 06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A  
gtecybertrustca, May 10, 2002, trustedCertEntry,  
Certificate fingerprint (MD5): C4:D7:F0:B2:A3:C5:7D:61:67:F0:04:CD:43:D3:BA:58  
gtecybertrust5ca, May 10, 2002, trustedCertEntry,  
Certificate fingerprint (MD5): 7D:6C:86:E4:FC:4D:D1:0B:00:BA:22:BB:4E:7C:6A:8E  
verisignclass2ca, Jun 29, 1998, trustedCertEntry,  
Certificate fingerprint (MD5): EC:40:7D:2B:76:52:67:05:2C:EA:F2:3A:4F:65:F0:D8

## Add a client certificate to the web tier

You can add a client certificate to your web tier to support client host validation or trusted sign-on.

**Note:** The following procedure assumes that you have already generated or obtained a client certificate for your web tier and imported the client certificate into a keystore.

1. Log on to the web tier system.
2. Stop the web tier web application server.
3. Copy the keystore containing the client certificate to one of two locations:
  - The web tier's web application WEB-INF folder
  - A network share accessible to your web tier
4. Open the web configuration file (`web.xml`) in a text editor, and perform the following steps:
  - a. Set the **keystore** parameter to the path of the keystore containing the web tier's certificate.
  - b. Set **customize-folder** to a folder on the web tier host.
  - c. Create an empty `webtier.properties` file in the `<Customize-Folder>/config` directory. You will specify the keystore password in the `webtier.properties` file later.

**Note:** The **keystorePassword** parameter has been removed from the web tier configuration file (`web.xml`) since Service Manager 9.34p2. You must enter your web client keystore password in a `webtier.properties` file located in the `<Customize-Folder>/config` directory.

- d. Save the `web.xml` file.
5. In the `webtier.properties` file, set the **keystorePassword** parameter to the password to access the client keystore. For details, see ["Encryption of client keystore passwords" on page 13](#).
6. Restart your web tier web application server.

## Update the cacerts keystore file

### Applies to User Roles:

System Administrator

If you use a private certificate authority to generate an SSL server certificate, you can add the private certificate authority to the list of trusted certificate authorities that exist in the Java cacerts keystore file. Sun distributes this file with JSSE and with JDK version 1.4.x and later releases. You can then distribute this updated cacerts file to your HP Service Manager clients so that they can validate the server's signed certificate.

**Note:** This procedure requires that you install a Java SDK of version 1.4.x or later on the server where you installed your private certificate authority.

To update the cacerts keystore file:

1. Log on to server where you installed your private certificate authority.
2. Open the operating systems command prompt.
3. Change directories to the Java SDK `bin` folder.
4. Type the following command to import your private certificate authority's certificate (for example, `ca-cert.pem`) into the Java cacerts file that you publish to the rest of your network. Change the path and variables as necessary.  

```
keytool -import -keystore ./cacerts -trustcacerts -file ca-cert.pem -storepass changeit
```
5. When keytool prompts you, type `y` to trust the private certificate authority's certificate.



## Use keytool to create a certificate request

### **Applies to User Roles:**

System Administrator

You can use the keytool utility provided with the Sun Microsystems™ Java Development Kit to create a certificate request.

To use keytool to create a certificate request:

1. Open your operating system's command prompt.
2. Change directories to your JDK's bin folder.
3. Type the following command to create a client certificate request.

```
keytool -certreq -alias clients -keystore clientcerts -file smwebtier_  
certrequest.crs
```

You can define your own names for the -alias, -keystore, and -file parameters. The names above are examples.

4. When the keytool utility prompts you, type the Keystore password.  
Keytool exports the client certificate key to the file and path you specified with the -file parameter.

## Use keytool to create a private key

### **Applies to User Roles:**

System Administrator

You can use the keytool utility provided with the Sun Microsystems™ Java Development Kit to produce a private key in a keystore.

To use keytool to create a private key:

1. Open your operating system's command prompt.
2. Change directories to your JDK's bin folder.
3. Type the following command to create a client private key.

```
keytool -genkey -alias clients -keystore clientcerts
```

You can define your own names for the -alias and -keystore parameters. The names above are

examples.

4. When the keytool utility prompts you, type the following information:
  - Keystore password
  - Fully qualified domain name of host
  - Organizational unit
  - Name of organization
  - City or locality
  - State or province
  - Two letter country code
5. Review the contact information and type yes to accept it.
6. Type in the password you want to use for the client key.

**Tip:** You can press ENTER to use the same password for the key as you typed for the keystore.

## Enable SSL encryption for external Web Services

If you want to connect to external Web Services using an SSL connection, you must import the CA certificate that signed the external Web service certificate into the CA certificate keystore for the Service Manager server. If you purchased a server certificate, import the external Web services CA certificate in the CA certificate keystore provided with your purchased certificate. If you generated your own server certificate by using a self-signed private CA certificate, then import the external Web Services CA certificate into your private CA certificate keystore instead.

**Note:** In this SSL scenario, the Service Manager server acts as a client of the external web service and must present the external web service with a valid certificate. If the external Web Service requires client authentication, you cannot use the Service Manager sample server certificate because the external web service will not recognize the signing authority of the sample certificate, and because the sample certificate uses a common name (CN) for the Service Manager host which will not match your actual Service Manager host name. HP recommends you purchase or create a valid certificate for the Service Manager host in order to establish an SSL-encrypted connection with external Web Services.

1. Import the CA certificate of the external Web Service into the CA certificate file of the Service Manager system. You may use a tool like keytool to import the external Web Service's CA certificate.
2. Stop the Service Manager server.
3. Open the Service Manager initialization file (`sm.ini`) with a text editor.
4. Add the following parameters to require SSL encryption using the external Web Service's signed certificate.
  - `keystoreFile` – Identify the keystore file containing the Service Manager server's certificate and private key
  - `keystorePass` – Identify the password to the keystore file containing the Service Manager server's certificate and private key
  - `truststoreFile:cacerts` – Identify the keystore file containing the external Web Service's CA certificate.
  - `truststorePass` – Identify the password to the keystore file containing the external Web Service's CA certificate
5. Save the Service Manager initialization file.
6. Restart the Service Manager server.
7. Login to the Service Manager system with an administrator.
8. Click **Tailoring > Web Services > Run WSDL to JS**.
9. Update the endpoint URL to the external Web Service to include the HTTPS protocol. For example, `https://remote_server.remote_domain.com:13445/remote_service.wsdl`.

**Note:** The endpoint URL must use the same server name as the common name (CN) listed in the external server certificate. For example, if the certificate uses the name `remote_server.remote_domain.com`, then the endpoint URL must also use the name `remote_server.remote_domain.com`.
10. Complete the wizard to connect to the external Web Service and convert it's WSDL into local JavaScript proxy code.

## Enable SSL encryption for published Web Services

If you want external Web Services clients to use an SSL connection with the Service Manager server, you must provide them with the CA certificate for the Service Manager server. If you purchased a server certificate, copy the CA certificate from the CA certificate keystore provided with your purchased certificate. If you generated your own server certificate by using a self-signed private CA certificate, copy the CA certificate from your private CA certificate keystore instead.

**Note:** HP recommends you do not use the Service Manager sample server CA certificate because the sample certificate uses a canonical name (CN) for the server which will not match your actual server name. The best practice is to purchase or create a valid certificate for the Service Manager server in order to establish an SSL-encrypted connection with external web service clients.

1. Copy the keystore that contains the CA certificate that signed your server's certificate, and send it to the systems running the external Web services clients. Out-of-box, the Service Manager uses a sample CA certificates keystore as part of the web tier.

**Note:** HP recommends using a CA certificate that you created or purchased instead of the default Service Manager CA certificate.

2. Import the CA certificate of the Service Manager system into the CA certificate keystore of the external Web Services client. You may use a tool like keytool to import Service Manager's CA certificate.
3. Configure the external Web Services client to use the update CA certificate keystore. Follow the instructions for your Web Services client to set the path to the CA certificate keystore.
4. Update the endpoint URL the external Web Services clients use to include the HTTPS protocol. For example, `https://myserver.mydomain.com:13443/SM/7/ws`. Follow the instructions for your Web Services client to update the endpoint URL.

**Note:** The endpoint URL must use the Service Manager server's common name (CN) as defined in the server certificate. For example, if the server certificate uses the name `myserver.mydomain.com`, then the endpoint URL must also use the name `myserver.mydomain.com`.

**Note:** If you want the external Web Service clients to download the Service Manager Web Service WSDL, point them to a URL using the following format:

`https://myserver.mydomain.com:13443/SM/7/<Service Name>.wsdl`

## What are PEM files?

Privacy Enhanced Mail (PEM) files are a type of Public Key Infrastructure (PKI) file used for keys and certificates. PEM, initially invented to make e-mail secure, is now an Internet security standard. HP Service Manager uses OpenSSL libraries to encrypt and decrypt SOAP messages over HTTP and requires certificates and keys in PEM format. The typical PEM files are:

- key.pem contains the private encryption key
- cert.pem contains certificate information

Because it is a standard, any PKI implementation can use .pem files as a repository for keys or certificates. OpenSSL supports a variety of standard formats in addition to .pem, including Distinguished Encoding Rules (DER) and X.509. OpenSSL has several utility functions that can convert these formats.

## What is a cacerts file?

The cacerts file is a collection of trusted certificate authority (CA) certificates. Sun Microsystems™ includes a cacerts file with its SSL support in the Java™ Secure Socket Extension (JSSE) tool kit and JDK 1.4.x. It contains certificate references for well-known Certificate authorities, such as VeriSign™. Its format is the "keystore" format defined by Sun. An administrator can edit the cacerts file with a command line tool (also provided by Sun) called keytool. For more information about keytool, see the Sun Web site.

**Note:** The default password for the cacerts file supplied by Sun is `changeit`. You must use this password to view the contents or to import a new certificate. For security reasons, change the default password.

The essential requirement is that the certificate authority that signed the HP Service Manager server's certificate must be in the list of certificate authorities named in this file. To use a self-issued server certificate created with OpenSSL or a tool such as Microsoft Certificate Server™, you must import the certificate for this private certificate authority into the cacerts file that the client uses for SSL. If you do not import the certificate, SSL connections fail because the Java SSL implementation does not recognize the certificate authority.

## Windows client error: No trusted certificate found

Users running the Windows client from a redeployed installer may experience the following error.

Error message	Cause and Solution	Affected application or component
<p>Error Connecting to server, server not up or wrong connection parameters: https://&lt;hostname&gt;:&lt;port&gt;</p> <p>Cause: javax.xml.soap.SOAPEException: Message send failed – sun.security.validator.ValidatorException: No trusted certificate found</p>	<p><b>Cause:</b> The path to the CA certificates file on the local Windows client installation is different from the path that the Client Packaging Utility originally saved. This is usually the result of the local user saving the Windows client to a different path than the Windows client used by the Client Configuration Utility.</p> <p><b>Solution:</b> Change the path to the CA Certificate file.</p> <ol style="list-style-type: none"> <li>1. Log on to the machine running the Windows client using a local administrator account.</li> <li>2. Start the Windows Client. The Windows client displays the "No trusted certificate found" error message.</li> <li>3. Click <b>OK</b>.</li> <li>4. Close the Connections window.</li> <li>5. Click <b>Window &gt; Preferences &gt; HP Service Manager &gt; Security</b>.</li> <li>6. In the <b>CA Certificates File</b> field, type or select the path to the CA certificates file.</li> </ol>	<p>Windows client</p>

## TLS 1.2 Support and Configuration

When the Service Manager web tier, the Mobility client, and SRC act as TLS/SSL servers, they use the Oracle or IBM Java virtual machine (JVM) that is required by your third-party web applications server to securely manage and connect incoming client requests that use TLS or SSL protocols. Examples of client requests include requests that originate from web browsers that are running on end-user's workstations. Depending on the third-party web application server, these connections default to the highest supported version of the TLS/SSL protocol. HP does not provide the software that hosts the Service Manager web tier, Mobility client, or SRC products; and it is this software that controls the specific TLS/SSL protocols that are used. As such, HP recommends that you consult the vendor of your browser and third-party web application server (such as IBM Websphere, Oracle WebLogic, and Apache Tomcat) for information about how to configure TLS 1.2.

As of Service Manager 9.41, the web tier, Mobility client and Windows client use TLS 1.2 by default.

The Service Manager server uses Java to securely manage and connect incoming client requests that use TLS and SSL protocols. In this scenario, the Service Manager server is acting as a TLS/SSL server. As of Service Manager 9.41, the Service Manager server uses TLS 1.2 by default (TLS 1.0 and TLS 1.1 are also supported). To enforce the TLS 1.2 protocol, configure the following parameter in the sm.ini file:

*sslProtocols:TLSv1.2*

When the Service Managerserver acts as a TLS/SSL client, it uses OpenSSL to connect to the Directory Services server via the secure LDAP protocol. Secure LDAP is also known as LDAP over SSL (LDAPS). Depending on the Directory Service server, these connections will default to the highest supported version of the TLS/SSL protocol. To force the Service Manager server to establish only TLS 1.2 connections with the LDAP server, set the following environment variable in the operating system:

`LDAPTLS_PROTOCOL_MIN=3.3`

**Note:** For information about how to set an environment variable, refer to your operating system's documentation or to your server administrator.

The Service Manager server may act as a TLS/SSL client to securely connect to SMTP servers or to consume third-party external Web Services over HTTPS, depending on the SMTP Server or the Web Services server. These connections will default to TLS 1.2. To enforce the TLS 1.2 protocol for these scenarios, please configure the following parameter in the sm.ini file:

*sslProtocols:TLSv1.2*

## Trusted sign-on

You can configure HP Service Manager clients to automatically log on using the same authentication information as users entered when they logged onto their client workstation's operating system. When you enable trusted sign-on, users bypass the Service Manager logon screen and directly enter the application.

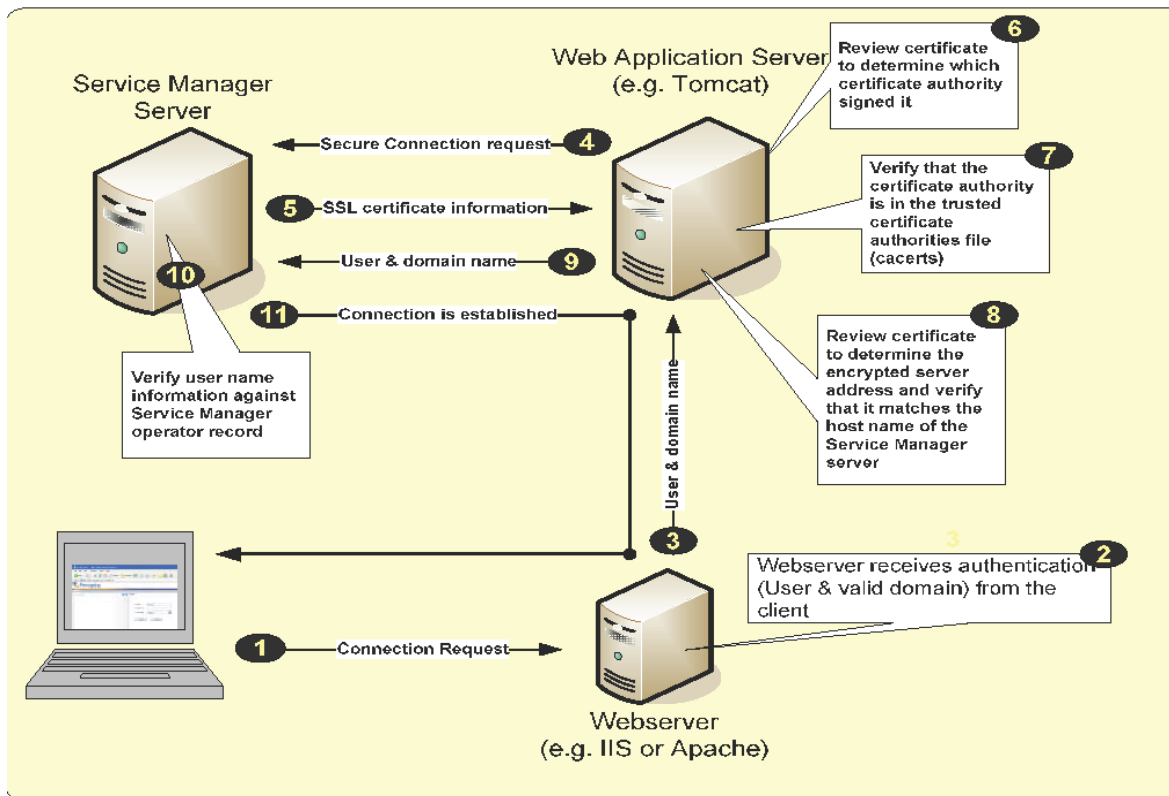
In a trusted sign-on scenario the Service Manager server grants access to clients only if the following conditions are met.

- The user's logon credentials match an existing operator record in Service Manager or a valid LDAP source that Service Manager recognizes.
- A trusted authentication authority, such as the operating system, validates that the user's logon credentials are valid.
- The client (Service Manager Web Tier or Windows) must present a signed SSL certificate.

The following figure depicts the connection process between a Web server, a Web application server, and the Service Manager application server:

- The Web server receives the user information from the client via the browser, and passes the user name and domain name to the Web application server.
- The Web application server (such as Tomcat, WebSphere®, or WebLogic Server®) acts as a client, and communicates with the Service Manager application server.
- The Service Manager application server also checks whether the user was authenticated by a valid domain. Local machine authentication is not accepted; if attempted the Service Manager server will reject such a request.





## Requirements for trusted sign-on

This configuration is intended for customers who have an HP Service Manager server running in a secured environment and want users to be able to log on to the server without providing a user name and password.

### Parameter required in the server configuration file (sm.ini)

```
trustedsignon:1
ssl:1
ssl_reqClientAuth:2
```

### Parameters required in the web tier configuration file (web.xml)

**isCustomAuthenticationUsed** – Set the value to false to make Service Manager send the current user name in the HTTP header.

### Other requirements

1. Configure your web application server to use a trusted authentication source.
  - a. Stop the web application server running the web tier.
  - b. Navigate to the folder <web tier installation path>\WEB-INF\classes, and open the file application-context.xml in a text editor.
  - c. Search for the following string:

```
/**=httpSessionContextIntegrationFilter,anonymousProcessingFilter
```
  - d. Replace the search string with the following text to use Microsoft Integrated Windows Authentication as your trusted authentication source:

```
/**=httpSessionContextIntegrationFilter,preAuthenticationFilter,anonymousProcessingFilter
```
  - e. Save the file and restart the web application server.
2. Create an operator record for each Windows user you want to log on to Service Manager. These operators do not need a password.
3. Install and configure an external authentication source, such as Microsoft Integrated Windows Authentication, to ensure that Service Manager can use your private certificates.
4. Beginning with Service Manager 9.30, Trusted Sign-On requires the parameter `ssl_reqClientAuth:2` to be set. You must then create unique client SSL certificates for each Service Manager client wanting to access Service Manager with Trusted Sign-On. For example, if you have 20 Service Manager Windows clients, you must create 20 unique client SSL certificates. If you have four Service Manager Web Tier servers, you must create four unique client SSL certificates. Note that if maintaining these unique client SSL certificates incurs unsustainable IT operation costs, you can consider the use of the `acceptsharedcert:1` parameter. See [Parameter: acceptsharedcert](#).

## Example: Enabling trusted sign-on

Trusted sign-on allows users on trusted clients who have logged into a Windows domain to log on to Service Manager without providing a user name and password. Trusted sign-on requires the web application server to connect to a web server (such as Windows Internet Information Services (IIS) or Apache http server) for third-party authentication.

### Notes:

- Service Manager 9.30 or later only supports trusted sign-on with SSL enabled and the `ssl_reqClientAuth` parameter set to "2".
- To use trusted sign-on, you must first add your web tier and Windows clients to a domain.

This example assumes that you are using Tomcat as the web application server and Apache or IIS as the web server.

To enable trusted sign-on, perform the following tasks:

## Task 1: Enable required SSL encryption and trusted clients.

For detailed steps, see ["Example: Enabling required SSL encryption and trusted clients" on page 69.](#)

## Task 2: Configure the web tier to use trusted sign-on.

1. Stop the web application server running the web tier.
2. In the web tier's `web.xml` file, set `isCustomAuthenticationUsed` to `false`.
3. In the `<Tomcat>/conf/server.xml` file, insert **`tomcatAuthentication="false"`** in the following section as shown below.

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

4. Edit the web application server's `application-context.xml` file to enable pre-authentication.
  - a. Open `<web tier installation path>\WEB-INF\classes\application-context.xml` in a text editor, and search for the following string:

```
/**=httpSessionContextIntegrationFilter,anonymousProcessingFilter
```

- b. Replace the search string with the following text:

```
/**=httpSessionContextIntegrationFilter,
preAuthenticationFilter,anonymousProcessingFilter
```

**Note:** If you need to enable trusted sign-on for your web client users and also enable web tier lightweight single sign-on (LW-SSO) for integrations, add `lwSsoFilter` followed by `preAuthenticationFilter`, as shown in the following:

```
/**=httpSessionContextIntegrationFilter,  
lwSsoFilter,preAuthenticationFilter,anonymousProcessingFilter
```

For information about how to enable LW-SSO in the web tier, see [Configure LW-SSO in the Web tier](#).

5. Restart the web application server.

## Task 3: Configure each Windows client to use trusted sign-on.

Do the following for each Windows client:

1. Make sure SSL encryption is enabled for the Windows client. See task 1.
2. Open a client connection, and on the Connection tab select **Use Trusted Sign-on**, and click **Apply**.

## Task 4: Install and configure the web server (Apache or IIS) to use trusted sign-on.

Install and configure an external authentication source, such as Microsoft Integrated Windows Authentication (IIS) or Apache, to ensure that Service Manager can use your private certificates. When using IIS, you need to configure an ISAPI connector for your web application server, and you need to modify the virtual directory to use Integrated Windows Authentication. For details, see "[Example: Configuring the web server for trusted sign-on](#)" on page 99.

## Task 5: Create an operator record for each Windows user.

Create an operator record for each Windows user you want to log in to Service Manager. The operator's login name must match the user's NT account username, but does not require a password.

## Task 6: Configure web browsers to enable web client users to use trusted sign-on.

Configure the web browser's security settings on each web client host. The following steps use Internet Explorer as an example.

1. Open Internet Explorer, and select **Tools > Internet Options**.
2. On the **Security** tab, click **Custom Level**, scroll down to the User Authentication section at the bottom, and select **Automatic logon with current username and password**.
3. On the **Security** tab, click **Trusted Sites > Sites**, and add the web tier's server address (FQDN) to the list of sites: `http://<myWebtierHostName>.<myDomain>`

**Note:** On FDCC-compliant computers, the security settings of Internet Explorer are locked by default and you cannot change them. For a workaround for this issue, see [Troubleshooting: Web client failed to automatically log in to Service Manager](#).

## Task 7: Test your trusted sign-on setup.

1. Start the Service Manager server, the web server (Apache or IIS), and the web application server (in this example, Tomcat).
2. Start a Windows client, and log in using trusted sign-on.

Service Manager should automatically log you in with your NT account username.

3. Start Internet Explorer, and open the web tier login URL:  
`http://<myWebtierHostName>.<myDomain>:<port>/webtier-x.xx//index.do`

Service Manager should automatically log you in without displaying the login screen.

## Example: Generating a server certificate with OpenSSL

The following example describes how to create a signed server certificate using the OpenSSL toolkit as a private certificate authority. This example also uses the keytool utility available with the Sun Microsystems™ standard Java Development Kit. You can use a server certificate to enable SSL encryption from the HP Service Manager server.

**Note:** The information contained in this example regarding OpenSSL technology is provided by HP as a courtesy to our customers and partners. This documentation does not replace an OpenSSL reference, and HP encourages you to conduct additional research regarding OpenSSL technology by consulting with sources outside of this document. HP hereby disclaims all liability associated with the use and accuracy of this information. As OpenSSL technology evolves, HP may or may not update this reference.

1. Obtain and install a Java platform on the server you want to use as your private certificate authority. See the Sun Microsystems™ Java Technology site for the necessary software and documentation.
2. Obtain and install OpenSSL on the server you want to use as your private certificate authority. See the OpenSSL Web site for the necessary software and documentation.

**Caution:** HP strongly recommends that you do not install your private certificate authority on the same server as your Service Manager production server.

3. Create and configure an `openssl.conf` file in the `bin` folder of your OpenSSL installation.
4. Open the operating system's command prompt on the private certificate authority server.
5. Change directories to the OpenSSL `bin` folder.
6. Type the following command to create the private key for your private certificate authority:  
`openssl genrsa -des3 -out cakey.pem 2048`
7. When OpenSSL prompts you, type the password phrase you want to use to protect your certificate authority's private key file (`cakey.pem`). For example, `CAKeyPassword`.  
You must use the same password phrase each time you sign a certificate request with your private certificate authority. If you forget this password, you must repeat the steps to create another certificate authority private key.
8. Type the following command to create a public certificate for your private certificate authority:  
`openssl req -new -key cakey.pem -x509 -days 1095 -out mycacert.pem -config .\openssl.conf`
9. Change directories to the Java platform's `bin` folder.
10. Type the following command to import your private certificate authority's certificate into the Java `cacerts` file that you will publish to the rest of your network.  
`keytool -import -keystore ./cacerts -trustcacerts -file mycacert.pem -storepass changeit`
11. When `keytool` prompts you, type `y` to trust the private certificate authority's certificate.

12. Install the updated Java cacerts file on the Service Manager server. Copy the `cacerts` file to the `RUN` folder of the Service Manager server. You can also copy the `cacerts` file to the `JRE\lib\security` folder of your Java Run Time Environment (RTE) or Java Development Kit (JDK).
13. Install the updated Java cacerts file on the Service Manager web tier. Copy the `cacerts` file to the `<web application server installation path>\WEB-INF` folder of the Service Manager web tier.
14. Install the updated Java cacerts file on the Service Manager Windows client. Copy the `cacerts` file to the `<Windows client installation path>\plugins\com.hp.ov.sm.client.common_x.xx` folder of the Service Manager Windows client.
15. Change directories to the Java platform's `bin` folder.
16. Type the following command to create a private key and keystore for your Service Manager server.  

```
keytool -genkey -alias myserver -keystore servercert.keystore
```
17. When `keytool` prompts you, type the password phrase you want to use to protect your Service Manager server's keystore file. For example, `SMServerKeyPassword`.

**Note:** Do not use the same password as your private certificate authority key password.

18. When `keytool` prompts you for your first and last name, type the fully qualified host name of your Service Manager server system.

**Note:** This is the same name your clients will use to connect to the Service Manager server.

19. When `keytool` prompts you for the organization unit, organization, city or locality, state or province, and two-letter country code, type the identification information for your company.
20. Verify the information you provided and type `yes` if it is correct.
21. When `keytool` prompts you for the password phrase to use for your Service Manager server's private key, press `ENTER` to use the same password as you created for the keystore.

**Note:** The password for the private key must match the password for the keystore file.

22. Type the following command to create a certificate request for your Service Manager server. For example, to create a certificate request for your HP Service Manager server type:  
  

```
keytool -certreq -alias myserver -keystore servercert.keystore -file smserver_certrequest.crs
```
23. When keytool prompts you, type the password for the HP Service Manager server's keystore file (from step 17). For example, `SMServerKeyPassword`.
24. Copy the Service Manager server's certificate request (For example, `smserver_certrequest.crs`) to the OpenSSL bin folder.

25. Change directories to the OpenSSL bin folder.

26. Type the following command to sign the Service Manager server's certificate request with your private certificate authority:  
  

```
openssl x509 -req -days 365 -in smserver_certrequest.crs -CA mycacert.pem -CAkey cakey.pem -CAcreateserial -out smserver_cert.pem
```

27. When OpenSSL prompts you, type the password for your certificate authority's private key. For example, `CAKeyPassword`.

OpenSSL stores the new signed certificate (`smserver_cert.pem`) in the `newcerts` directory.

**Tip:** To view the contents of the signed certificate, you can type following command:

```
openssl x509 -in smserver_cert.pem -text -noout
```

28. Copy the signed client certificate (`smserver_cert.pem`) to the OpenSSL server's Java platform bin folder.
29. Open the operating system's command prompt.
30. Change directories to the Java platform's bin folder.
31. Type the following command to import the Service Manager server's signed certificate into the server keystore.  
  

```
keytool -import -trustcacerts -alias myserver -keystore ./servercert.keystore -file smserver_cert.pem
```
32. When keytool prompts you to trust the private certificate authority's certificate, type `y`.



33. When keytool prompts you, type the password to your server's keystore file. For example, `SMServerKeyPassword`.

## Example: Generating a client certificate with OpenSSL

The following example describes how to create a signed client certificate using the OpenSSL toolkit as a private certificate authority. This example also uses the keytool utility available with the Sun Microsystems™ standard Java Development Kit. You can use a client certificate to validate that the client is authorized to connect to HP Service Manager server or as part of a trusted sign-on configuration.

**Note:** This example builds on information presented in ["Example: Generating a server certificate with OpenSSL" on page 93](#). The information contained in this example regarding OpenSSL technology is provided by HP as a courtesy to our customers and partners. This documentation does not replace an OpenSSL reference, and HP encourages you to conduct additional research regarding OpenSSL technology by consulting with sources outside of this document. HP hereby disclaims all liability associated with the use and accuracy of this information. As OpenSSL technology evolves, HP may or may not update this reference.

1. Log on to server where you installed your OpenSSL private certificate authority, and open the operating system's command prompt.
2. Change directories to the Java platform's bin folder.
3. Type the following command to create a private key and keystore for your Service Manager client. For example, to create a private key and keystore for your Service Manager web tier, type:  
`keytool -genkey -alias clients -keystore <clientcerts>.keystore`

**Note:** When you repeat this step for multiple clients, replace `<clientcerts>` (and also `<client>` in the following steps) with a name that can identify every single client. For example, you can use the FQDN for each Windows client, and use the FQDN or smwebtier for your web tier client.

4. When keytool prompts you, type the password phrase you want to use to protect your Service Manager client's keystore file. For example, `ClientKeyPassword`.
5. When keytool prompts you for your first and last name, type the fully qualified host name of your Service Manager client system.

6. When keytool prompts you for the organization unit, organization, city or locality, state or province, and two-letter country code, type the identification information for your company.
7. Verify the information you provided and type *yes* if it is correct.
8. When keytool prompts you for the password phrase to use for your Service Manager web tier's private key, press ENTER to use the same password as you created for the keystore.

**Note:** The password for the private key must match the password for the keystore file.

9. Type the following command to create a certificate request for your Service Manager client. For example, to create a certificate request for your Service Manager web tier, type:  

```
keytool -certreq -alias clients -keystore <clientcerts>.keystore -file  
<client>_certrequest.crs
```
10. When keytool prompts you, type the password for the Service Manager client's keystore file (from step 4). For example, *ClientKeyPassword*.
11. Copy the Service Manager client's certificate request (For example, *<client>\_certrequest.crs*) to the OpenSSL bin folder.
12. Change directories to the OpenSSL bin folder.
13. Type the following command to sign the Service Manager client's certificate request with your private certificate authority:  

```
openssl x509 -req -days 365 -in <client>_certrequest.crs -CA mycacert.pem -  
CAkey cakey.pem -CAcreateserial -out <client>_cert.pem
```
14. When OpenSSL prompts you, type the password for your certificate authority's private key. For example, *CAKeyPassword*.

OpenSSL stores the new signed certificate (*<client>\_cert.pem*) in the *newcerts* directory.

**Tip:** To view the contents of the signed certificate, you can type following command:

```
openssl x509 -in <client>_cert.pem -text -noout
```

15. Copy the signed client certificate (*<client>\_cert.pem*) to the OpenSSL server's Java platform bin folder.
16. Open the operating system's command prompt.

17. Change directories to the Java platform's bin folder.
18. Type the following command to import the Service Manager client's signed certificate into a client keystore.

```
keytool -import -trustcacerts -alias clients -keystore ./<clientcerts>.keystore  
-file <client>_cert.pem
```
19. When keytool prompts you to trust the private certificate authority's certificate, type y.
20. Copy the updated client keystore (<clientcerts>.keystore) to the default certificate path of your client:
  - WEB-INF folder of the Service Manager Web tier
  - *<Windows client installation path>\plugins\com.hp.ov.sm.client.common\_x.xx* folder of your Service Manager Windows clients
21. If you are using a trusted clients or trusted sign-on implementation, do the following:
  - a. Import each client certificate you want to be part of the list of trusted clients to a trusted clients keystore. To do so, type the following command:

```
keytool -import -alias client1 -file <client>_cert.pem -keystore  
trustedclients.keystore
```
  - b. Copy the trusted clients keystore (trustedclients.keystore) to the Service Manager server's RUN folder.

## Example: Configuring the web server for trusted sign-on

To enable trusted sign-on for web tier clients, you must install and configure a web server (for example, Microsoft Internet Information Services (IIS) or Apache), which receives the user information from the client via the browser, and passes the user name and domain name to the web application server.

**Note:** The following steps assume that the web server and web application server configurations are already established, and that the only necessary changes to the configurations of these servers are those described below.

### Apache configuration changes

**Note:** The mod\_auth\_sspi.so module is available only for Windows; if Apache is installed on a UNIX® operating system, it may be necessary to create a custom class to perform trusted sign-on.

1. Add the `mod_auth_sspi.so` module to the `/modules` directory in the Apache installation.
2. Add the following lines to the bottom of the `httpd.conf` file to allow for trusted sign-on:

```
#SspiAuth Module
LoadModule sspi_auth_module modules/mod_auth_sspi.so
<Location "/sm">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
    AuthType SSPI
    SSPIAuth On
    SSPIDomain MYDOMAIN
    SSPIAuthoritative On
    SSPIOfferBasic Off
    SSIPerRequestAuth On
    require valid-user
</Location>
```

The name within the Location tag needs to be the path the user enters to open the Service Manager web client Web site; it is usually `/sm`, because the name is taken from the `sm.war` file. In a configuration with multiple domains, comment out the `SSPIDomain` parameter by adding a crosshatch character (`#`) in front of the line.

## IIS configuration changes

**Note:** The following steps are for IIS 5 only. The steps for other versions of IIS may vary.

1. Download the Apache jk2 connector binary from [jakarta.apache.org](http://jakarta.apache.org) to your web application server's home directory (in this example, Tomcat home directory), and make sure the following files are added to the following folders:
  - `<Tomcat_home>\conf` folder: `jakarta_isapirewrite.reg`, `jk2.properties`, `workers2.properties`
  - `<Tomcat_home>\bin` folder: `isapi_redirector2.dll`
2. In the `jakarta_isapirewrite.reg` and `workers2.properties` files, update the Tomcat root path to your real Tomcat path.
3. Run the `jakarta_isapirewrite.reg` file to add the information to your web tier server's system registry.

4. From your operating system's Control Panel, click **Administrative Tools > Internet Information Services**.
5. Expand your local computer node, click **Web Sites**.
6. Right-click **Default Web Sites**, and select **New > Virtual Directory**.

The Virtual Directory Creation Wizard opens.

7. In the Name field, type **jakarta**, and in the Directory field, browse to the directory where `isapi_redirector2.dll` is located (in this example, it is the Tomcat bin folder, with read and execute permissions).
8. Right-click **Default Web Sites**, and select **Properties**. In the Properties window, do the following:
  - a. Select the **ISAPI Filter** tab, and add a filter with the following information:
    - Filter Name: jakarta
    - Executable: `C:\<Tomcat_home>\bin\isapi_redirector2.dll`

**Note:** Replace `<Tomcat_home>` with your real Tomcat home directory.

- b. Select the **Directory Security** tab, and click **Edit**.

The Authentication Methods window opens.

- c. Select **Integrated Windows authentication** at the bottom, and clear all other selections in this window.
9. Start your operating system's command prompt, and run the following commands:

```
cd C:\Inetpub\AdminScripts
cscript adsutil.vbs set w3svc/NTAuthenticationProviders "NTLM"
```

Your IIS web server configuration changes are completed. You can continue to configure the web browser's security settings to enable trusted sign-on for web clients.

## Example: Viewing the contents of a cacerts file

To view the entries in a `cacerts` file, you can use the `keytool` utility provided with Sun J2SDK versions 1.4 or later. The following example uses the `-list` command to display the CA certificates in the `cacerts` file.

```
C:\j2sdk1.4.2_04\jre\bin>keytool -list -keystore ./cacerts
Enter keystore password:  changeit
```

Keystore type: jks  
Keystore provider: SUN

Your keystore contains 15 entries

```
thawtepersonalfreemailca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): 1E:74:C3:86:3C:0C:35:C5:3E:C2:7F:EF:3C:AA:3C:D9
thawtepersonalbasicca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): E6:0B:D2:C9:CA:2D:88:DB:1A:71:0E:4B:78:EB:02:41
baltimorecodesigningca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): 90:F5:28:49:56:D1:5D:2C:B0:53:D4:4B:EF:6F:90:22
verisignclass3ca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): 78:2A:02:DF:DB:2E:14:D5:A7:5F:0A:DF:B6:8E:9C:5D
gtecybertrustglobalca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): CA:3D:D3:68:F1:03:5C:D0:32:FA:B8:2B:59:E8:5A:DB
thawtepersonalpremiumca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): 3A:B2:DE:22:9A:20:93:49:F9:ED:C8:D2:8A:E7:68:0D
thawteserverca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): C5:70:C4:A2:ED:53:78:0C:C8:10:53:81:64:CB:D0:1D
verisignclass4ca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): 1B:D1:AD:17:8B:7F:22:13:24:F5:26:E2:5D:4E:B9:10
baltimorecybertrustca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): AC:B6:94:A5:9C:17:E0:D7:91:52:9B:B1:97:06:A6:E4
verisignclass1ca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): 51:86:E8:1F:BC:B1:C3:71:B5:18:10:DB:5F:DC:F6:20
verisignserverca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): 74:7B:82:03:43:F0:00:9E:6B:B3:EC:47:BF:85:A5:93
thawtepremiumserverca, Feb 12, 1999, trustedCertEntry,
    Certificate fingerprint (MD5): 06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A
gtecybertrustca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): C4:D7:F0:B2:A3:C5:7D:61:67:F0:04:CD:43:D3:BA:58
gtecybertrust5ca, May 10, 2002, trustedCertEntry,
    Certificate fingerprint (MD5): 7D:6C:86:E4:FC:4D:D1:0B:00:BA:22:BB:4E:7C:6A:8E
verisignclass2ca, Jun 29, 1998, trustedCertEntry,
    Certificate fingerprint (MD5): EC:40:7D:2B:76:52:67:05:2C:EA:F2:3A:4F:65:F0:D8
```

## Update the cacerts keystore file

### Applies to User Roles:

System Administrator

If you use a private certificate authority to generate an SSL server certificate, you can add the private certificate authority to the list of trusted certificate authorities that exist in the Java cacerts keystore file. Sun distributes this file with JSSE and with JDK version 1.4.x and later releases. You can then

distribute this updated cacerts file to your HP Service Manager clients so that they can validate the server's signed certificate.

**Note:** This procedure requires that you install a Java SDK of version 1.4.x or later on the server where you installed your private certificate authority.

To update the cacerts keystore file:

1. Log on to server where you installed your private certificate authority.
2. Open the operating systems command prompt.
3. Change directories to the Java SDK bin folder.
4. Type the following command to import your private certificate authority's certificate (for example, cacert.pem) into the Java cacerts file that you publish to the rest of your network. Change the path and variables as necessary.  

```
keytool -import -keystore ./cacerts -trustcacerts -file cacert.pem -storepass changeit
```
5. When keytool prompts you, type y to trust the private certificate authority's certificate.

## Use keytool to create a certificate request

### **Applies to User Roles:**

System Administrator

You can use the keytool utility provided with the Sun Microsystems™ Java Development Kit to create a certificate request.

To use keytool to create a certificate request:

1. Open your operating system's command prompt.
2. Change directories to your JDK's bin folder.
3. Type the following command to create a client certificate request.

```
keytool -certreq -alias clients -keystore clientcerts -file smwebtier_  
certrequest.crs
```

You can define your own names for the -alias, -keystore, and -file parameters. The names above are examples.

4. When the keytool utility prompts you, type the Keystore password.

Keytool exports the client certificate key to the file and path you specified with the `-file` parameter.

## Use keytool to create a private key

### **Applies to User Roles:**

System Administrator

You can use the keytool utility provided with the Sun Microsystems™ Java Development Kit to produce a private key in a keystore.

To use keytool to create a private key:

1. Open your operating system's command prompt.
2. Change directories to your JDK's bin folder.
3. Type the following command to create a client private key.

```
keytool -genkey -alias clients -keystore clientcerts
```

You can define your own names for the `-alias` and `-keystore` parameters. The names above are examples.

4. When the keytool utility prompts you, type the following information:
  - Keystore password
  - Fully qualified domain name of host
  - Organizational unit
  - Name of organization
  - City or locality
  - State or province
  - Two letter country code
5. Review the contact information and type `yes` to accept it.
6. Type in the password you want to use for the client key.

**Tip:** You can press ENTER to use the same password for the key as you typed for the keystore.



## Common Access Card (CAC) sign-on

Starting with version 9.32, the Service Manager web client supports CAC sign-on. System Administrators can configure the Service Manager server and web client to automatically log on using CAC authentication. CAC sign-on enables users to log in to the web client directly with a smart card that stores a valid user certificate, and users only need to enter a card PIN, instead of a user name and password.

**Note:** The Service Manager Windows client does not support CAC sign-on.

## Supported smart cards

During CAC sign-on, the Service Manager web tier gets access to the user authentication public certificate and its counterpart private key through the underlying client crypto architecture. In other words, Service Manager does not directly communicate with the card reader. Technically, Service Manager supports any smart cards that store an X.509 user authentication certificate and are designed to work with smart card middleware (such as ActivClient) that is installed on the user's computer. The term "Common Access Card (CAC)" in its origin refers to the United States Department of Defense (DoD) issued cards; however, in the Service Manager documentation the term CAC is used in a broader scope and refers to any of the supported smart cards. Two examples of such smart cards are the Common Access Card issued by DoD and Personal Identity Verification (PIV) cards issued by other US Government agencies.

## CAC authentication

A CAC smart card enables two-factor authentication: the CARD being something you have, and the CAC PIN being something you know. The CAC stores, in a secure way, the card holder's public certificates and its corresponding private keys. One of those stored certificates with its private keys is used for user authentication. Service Manager uses SSL and CAC authentication, which utilizes the certificate in the CAC card to identify the CAC card holder details and leverages the built-in SSL user authentication abilities to ensure that the CAC owner possesses the private key. The certificate key-pair proves that this person is who the person claims to be. For example, if the certificate claims the public key is owned by userX and it was signed by a Certificate Authority (CA), when the certificate is presented assuming Service Manager trusts the CA (which means the CA public key is put in Service Manager's trusted CA store), Service Manager would trust the information and then challenge the person presenting the

certificate to prove the person owns the matching private key. If the person provides the proper PIN, that private key is unlocked and then Service Manager authenticates the client as the one detailed in the certificate.

## Flexible mapping between distinguished name and login name

During CAC sign-on, a unique attribute in the user's certificate is retrieved as the user's Distinguished Name (DN) and sent to the SM server as the user's login name.

Different organizations may have different definitions for the same attribute in the user's certificate. For example, your organization may use employee ID for Subject.CN, while another may use user's full name for it. In addition, it is also possible that an organization needs to use an Subject Alternative Name (SAN) sub-attribute as DN. For this reason, the Service Manager web tier provides a configuration file (`<web tier installation path>\WEB-INF\classes\cacconf.properties`), which contains two parameters ("[certificateFieldExtractDN](#)" on page 114 and "[certificateOIDMapping](#)" on page 116) to allow flexible mapping between DN and login name. For more information, see "[Example: enabling CAC sign-on](#)" on page 111.

## Required configurations

CAC sign-on requires the following configurations:

- Both of the Service Manager server and web client have been configured to support CAC sign-on mode.
- Two-way SSL is configured between the web server (or the web application server if no web server) and browser.
- Two-way SSL is configured between the Service Manager server and all clients (the web and Windows clients, and web services clients such as SRC, Mobility, and other integrated products).

**Note:** Enabling CAC sign-on on the SM server side will affect all SM clients - all SM clients can only connect to the server through two-way SSL.

- JDK 1.6 or above is installed on the host machine running the web tier.

## Compatibility with other sign-on modes

The following table describes whether or not users can use other sign-on modes when CAC sign-on is enabled in the SM server and web client.

Mode	Description
Basic Authentication	<p><b>Web client:</b> Users can no longer log in to the web client by entering a user name and password (that is, using basic authentication). It is impossible that some web client users use CAC sign-on while some others use basic authentication. All users must use CAC sign-on.</p> <p><b>Windows client:</b> Users can log in to the Windows client only with basic authentication, but two-way SSL authentication is required between the server and Windows client.</p>
Trusted Sign-On (TSO)	When both CAC sign-on and TSO are configured in the same web tier, CAC sign-on takes precedence. For this reason, do not enable CAC sign-on and TSO in the same web tier at the same time.
Lightweight Single Sign-On (LW-SSO)	<p>The following discussion assumes a scenario where Product A (which does not support CAC access) is integrated with Service Manager and LW-SSO are enabled on both product sides.</p> <p><b>Outgoing LW-SSO:</b> SM web client users can directly access product A through LW-SSO. In other words, outgoing LW-SSO from the web tier works well.</p> <p><b>Incoming LW-SSO:</b> Incoming LW-SSO is ignored, and users can only access SM from product A with a CAC.</p>

## Conditions for CAC access

In a CAC sign-on scenario, the Service Manager server grants access to web clients only if the following conditions are met:

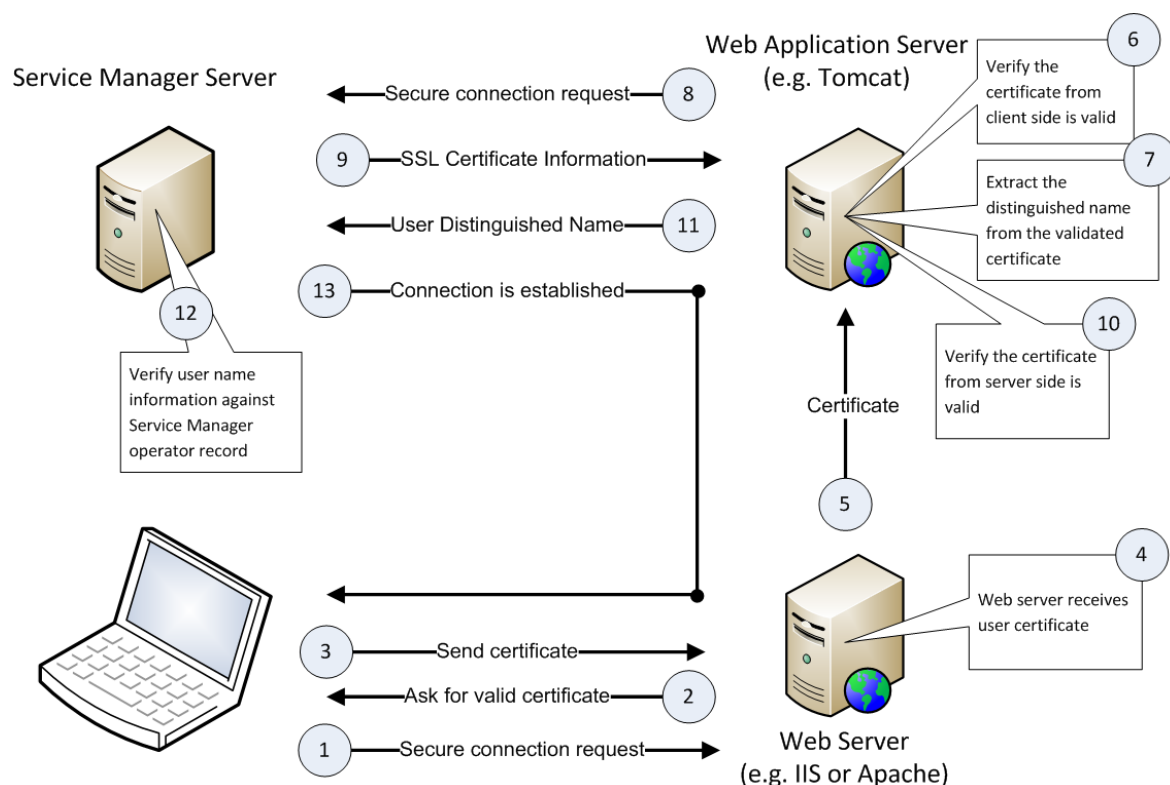
- The web browser from the client side sends the user certificate to the web server, and then the web server passes the certificate to the web application server.
- The user certificate passes the validation by the web tier.
- The user's logon credentials match an existing operator record in Service Manager or a valid LDAP source that Service Manager recognizes.

- The web client presents a signed SSL certificate.

**Note:** Each Windows client must also present a signed SSL certificate to the server.

## CAC connection process

The following figure depicts the connection process between a web server, a web application server, and the Service Manager server in a CAC sign-on scenario.



The following describes how the process works:

1. The user inserts a CAC into a card reader.
2. The smart card middleware (such as ActivClient) automatically loads the user's personal certificate ("user certificate") from the CAC to the browser.
3. The browser prompts the user to select an installed certificate to log in.
4. The user selects the user certificate to log in.

5. The browser prompts the user to enter the PIN of the user's CAC.
6. The user enters the correct PIN.
7. The web server passes the user certificate to the web application server for validation based on a pre-configured validation strategy, which can include any combinations of the following checks except that at least one of the following options must be included: using a local CRL, using an online CRL, and using the OCSP.
  - Check if the issuer is from a trusted CA (Certificate Authority)
  - Check if the certificate is revoked using a local Certificate Revocation List (CRL)
  - Check if the certificate is revoked using an online CRL
  - Check if the certificate is revoked using the Online Certificate Status Protocol (OCSP)
  - Check if the certificate has expired
  - Check if the certificate type is Smart Card
8. The web application server extracts the Distinguished Name (DN) from the user certificate and sends the DN as the user's user name to the Service Manager server.
9. The server verifies the DN against the `name` field in the `operator` table (when LDAP is not configured) or against the LDAP attribute mapped to this field (if LDAP is configured). If the DN value matches an operator name, the server logs the user in; otherwise the server rejects the login request.

## Requirements for CAC sign-on

This configuration is intended for customers who have an HP Service Manager server running in a secured environment and want users to be able to log on to the server using a Common Access Card (CAC), without the need to enter a user name and password.

## Parameters required in `sm.ini`

Service Manager 9.32 or later supports CAC sign-on with two-way SSL enabled and the `ssl_reqClientAuth` parameter set to either "1" or "2".

The following table lists the server parameters required for CAC sign-on.

ssl_reqClientAuth:1	ssl_reqClientAuth:2
cacsignon:1	cacsignon:1
ssl:1	ssl:1
sslConnector:1	sslConnector:1
ssl_reqClientAuth:1	ssl_reqClientAuth:2
keystoreFile:<servercert.keystore>	keystoreFile:<servercert.keystore>
keystorePass:<keystoreFile password>	keystorePass:<keystoreFile password>
truststoreFile:cacerts	truststoreFile:cacerts
truststorePass:<truststoreFile password>	truststorePass:<truststoreFile password>
<b>Note:</b> When using ssl_reqClientAuth:1, the ssl_trustedClientsJKS and ssl_trustedClientsPwd parameters are not required.	ssl_trustedClientsJKS:<trustedclients.jks>
	ssl_trustedClientsPwd:<ssl_trustedClientsJKS password>

**Note:** Once cacsignon is enabled (set to 1), three parameters are automatically (and implicitly) set to 1. See the following table.

Parameter	Value	Notes
ssl	1	Any other values explicitly specified in sm.ini are ignored.
sslConnector	1	
ssl_reqClientAuth	1	Any other values explicitly specified in sm.ini are ignored except for ssl_reqClientAuth:2.

## Parameters required in web.xml

The following parameters are required in the web tier configuration file (web.xml):

- isCustomAuthenticationUsed=false

Set the value to false to make Service Manager send the current user name in the HTTP header.

- `CACLogin=true`

Set the value to true to enable the CAC logon mode in the web tier.

## Other requirements

- Configure your web application server to enable CAC authentication. You do so by updating the web tier's `application-context.xml` file. For details, see ["Example: enabling CAC sign-on" below](#).
- When CAC logon is enabled in the server, you can set `ssl_reqClientAuth:1` or `ssl_reqClientAuth:2` in the `sm.ini` file. You must then create unique client SSL certificates for each Service Manager client wanting to access Service Manager with CAC. For example, if you have 20 Service Manager Windows clients, you must create 20 unique client SSL certificates. If you have 4 Service Manager Web Tier servers, you must create 4 unique client SSL certificates for them. In addition, you need to configure SSL in the web tier configuration file (`web.xml`) and also in the Windows client Preferences setting. For details, see ["Example: Enabling required SSL encryption and trusted clients" on page 69](#).

**Tip:** If maintaining these unique client SSL certificates incurs unsustainable IT operation costs, you can consider the use of the `acceptsharedcert:1` parameter, which enables all clients to use a "shared certificate". For more information, see ["Parameter: acceptsharedcert" on page 1](#).

- CAC sign-on requires two-way SSL connections between the web server (or web application server if no web server) and the user's browser. You need to set up two-way SSL on the web server, or on the web application server (if you have no web server deployed).
- Each CAC user must have an operator record created in Service Manager.

## Example: enabling CAC sign-on

This example assumes that you are using Tomcat as your web application server and Apache as your web server.

## Prerequisites

Before you proceed, make sure you have met the following prerequisites:

- Your organization has already a Common Access Card (CAC) security system in place; each Service Manager user owns a CAC card that stores the user's personal certificate ("user certificate") and you have downloaded your organization's CAC root CA certificate.

Your CAC root CA certificate refers to the root CA certificate that signs the user certificates stored in the CACs. Later, you will need to:

- copy it to your web tier host and specify its location in the `RootCertPath` parameter. See ["Task 3. Configure the web tier to use CAC sign-on." on the next page](#)
  - import it to your Tomcat trust store (this is required only when you have Tomcat deployed without Apache). See ["Task 6. Configure your web application server to use SSL." on page 121](#)
  - copy it to your Apache server and configure it in your Apache's SSL configuration file. See ["Task 7. Configure the web server to use SSL." on page 124](#)
  - let users import it to their web browser. See ["Task 9. Configure web browsers to enable web client users to use CAC." on page 127](#)
- You have Keytool and OpenSSL installed on a machine on which you will generate SSL certificates.

## Configuration Tasks

To enable CAC sign-on in your Service Manager system, complete the following tasks.

Task 1. Enable required SSL encryption and trusted clients.

Enable required SSL encryption between the SM server and Windows and web clients. For details, see ["Example: Enabling required SSL encryption and trusted clients" on page 69](#).

**Note:** This example uses `ssl_reqClientAuth:2`; however, CAC sign-on also allows the use of `ssl_reqClientAuth:1`. For more information, see ["Requirements for CAC sign-on" on page 109](#).

When the configuration is complete, verify that you can log in to the web client as usual and log in to Windows client with SSL encryption.

Task 2. Enable CAC sign-on in the SM server.

Set required parameters in the server configuration file (`sm.ini`). For details, see ["Requirements for CAC sign-on" on page 109](#).



Task 3. Configure the web tier to use CAC sign-on.

1. Stop the web application server running the web tier.
2. Copy your CAC root certificate (s) to a directory on your web tier host. For example:  
`<webtier>/WEB-INF/`
3. Set the following parameter values in the web tier's `web.xml` file.

Parameter	Value
<code>isCustomAuthenticationUsed</code>	<code>false</code>
<code>CACLogin</code>	<code>true</code>

4. Configure your web application server to use CAC as the trusted authentication source.
  - a. Navigate to the `<web tier installation directory>\WEB-INF\classes` folder, and open the `application-context.xml` file in a text editor.

- b. Search for the following string:

```
/**=httpSessionContextIntegrationFilter,anonymousProcessingFilter
```

- c. Replace the search string with the following text to use CAC authentication as your trusted authentication source:

```
/**=httpSessionContextIntegrationFilter,cacFilter,anonymousProcessingFilter
```

**Note:** If you also want to enable lightweight single sign-on (LW-SSO) for integrations, add `cacFilter` followed by `lwSsoFilter`, as shown in the following:

```
/**=httpSessionContextIntegrationFilter,  
cacFilter,lwSsoFilter,anonymousProcessingFilter
```

Be aware that CAC sign-on can only work with outgoing LW-SSO (LW-SSO from SM to another product). For information about how to enable LW-SSO in the web tier, see [Configure LW-SSO in the Web tier](#).

**Note:** If you enable both TSO and CAC sign-on in the same web tier, CAC sign-on takes precedence.

- d. Uncomment the following lines:

```
<bean id="cacFilter"
class="com.hp.ov.sm.client.webtier.cac.CACPreAuthenticationFilter">
    <property name="authenticationManager">
        <ref bean="authenticationManager"/>
    </property>
    <property name="defaultRole">
        <value>ROLE_PRE</value>
    </property>
    <property name="cacProps">
        <ref bean="cacConfiguration"/>
    </property>
</bean>

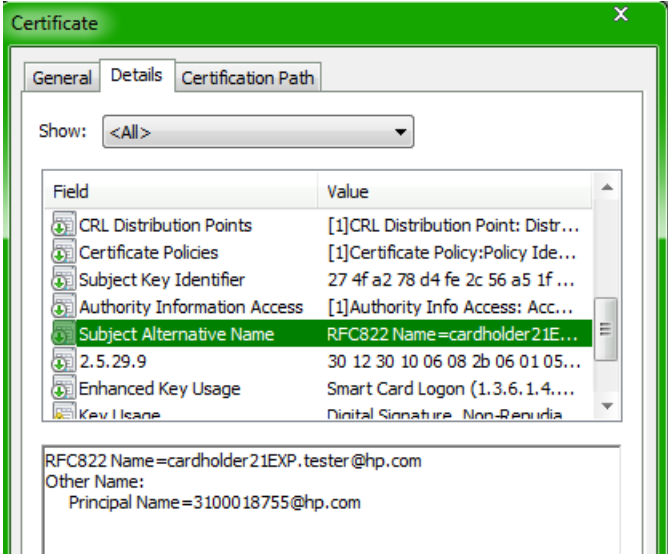
<bean id="cacConfiguration"
class="org.springframework.beans.factory.config.PropertiesFactoryBean">
    <property
name="location"><value>classpath:cacconf.properties</value></property>
</bean>
```

e. Save the file.

5. Configure CAC authentication in the web tier's cacconf.properties file.

- a. Open <web tier installation path>\WEB-INF\classes\cacconf.properties in a text editor.
- b. Specify the following parameters.

Parameter	Default	Description
certificateFieldExtractDN	Subject.CN	<p>This parameter specifies a unique attribute in the user certificate that the web application server extracts as a Distinguished Name (DN) and sends to the SM server as the user's user name.</p> <p>You can specify a Subject attribute in this parameter. By default, the Subject CN attribute (the certificate owner's Common Name) is extracted as the user's DN.</p>

Parameter	Default	Description
		<p>You can also specify this parameter as one of the following Subject Alternative Name (SAN) sub-attributes: RFC822 Name, and Other Name (see the figure below).</p> <div>The screenshot shows a 'Certificate' window with the 'Details' tab selected. A 'Show:' dropdown is set to '&lt;All&gt;'. A table lists certificate fields: CRL Distribution Points, Certificate Policies, Subject Key Identifier, Authority Information Access, Subject Alternative Name (highlighted), 2.5.29.9, Enhanced Key Usage, and Key Usage. Below the table, the 'Subject Alternative Name' details are shown: 'RFC822 Name=cardholder21EXP.tester@hp.com' and 'Other Name: Principal Name=3100018755@hp.com'.</div> <ul style="list-style-type: none"><li>• If using a sub-attribute of Other Name, specify the parameter in this format: SAN_OTHERNAME.&lt;Alias&gt;, where &lt;Alias&gt; is a name that you specify for the sub-attribute. The user certificate's SAN Other Name value that binds to the OID will be extracted as the DN. For example, if you use Other Name's Principal Name sub-attribute (also known as UPN, which should be mapped to OID "1.3.6.1.4.1.311.20.2.3"), you can specify the following value (where PrincipalName is an alias for Principal Name):  certificateFieldExtractDN=SAN_OTHERNAME.PrincipalName</li></ul> <div><p><b>Tip:</b> As a best practice, avoid including white spaces in alias names. For example, for Principal Name, specify PrincipalName or PN as an alias.</p><p><b>Note:</b> Next, you will specify a mapping between the alias and object ID (OID) of the specified sub-attribute of SAN Other Name, by using the <a href="#">"certificateOIDMapping" on the</a></p></div>

Parameter	Default	Description
		<p><a href="#">next page</a> parameter.</p> <ul style="list-style-type: none"> <li>If using the RFC822 Name sub-attribute, specify the following value: <code>certificateFieldExtractDN=SAN.RFC822</code></li> </ul> <p><b>Examples:</b></p> <p><code>certificateFieldExtractDN=Subject.E</code>  <code>certificateFieldExtractDN=SAN_OTHERNAME.PrincipalName</code></p>
certificateOIDMapping	(empty)	<p>This parameter specifies a mapping between the &lt;Alias&gt; you specified in <code>certificateFieldExtractDN</code> and an OID. Specify the parameter in this format: &lt;Alias&gt;:&lt;OID&gt;.</p> <p>For example, for SAN's Principal Name, specify this mapping:  <code>certificateOIDMapping=PrincipalName:1.3.6.1.4.1.311.20.2.3</code></p>
validationStrategy	(empty)	<p>This parameter specifies how the web application server verifies each user certificate.</p> <p>You can specify a combination of the following values, each of which represents a checking method:</p> <ul style="list-style-type: none"> <li>1: Check the revocation status of the user certificate using a local Certificate Revocation List (CRL).</li> <li>2: Check the revocation status of the user certificate using an online CRL.</li> <li>3: Check the revocation status of the user certificate using the Online Certificate Status Protocol (OCSP).</li> <li>4: Check if the user certificate is of the Smart Card type.</li> </ul> <p>Pay attention to the following:</p> <ul style="list-style-type: none"> <li>You cannot leave this parameter empty.</li> <li>If you specify more than one value, separate them with a</li> </ul>

Parameter	Default	Description
		<p>comma. In this case, all specified checking methods are used in the sequence they are listed in the parameter.</p> <ul style="list-style-type: none"> <li>You must specify at least one of the following values: 1, 2, and 3. If you do not do so, users will not be able to log in; instead, they will see a message in their browser that states "An error has occurred"; in the meantime, an error message will occur in the web application server's <code>sm.log</code> file.</li> <li>Smart Card checking (value 4) is mandatory for CACs, because CACs are smart cards.</li> <li>You can configure CRL or OCSP checking either in your web server or in this parameter, but not in both.</li> </ul> <p><b>Example:</b></p> <p><code>validationStrategy=3,4</code></p>
RootCertPath	(empty)	<p>This parameter specifies the file path (either absolute or relative) to the issuer's CA certificate.</p> <div> <p><b>Note:</b> The issuer's CA certificate refers to the root CA certificate that signs the user certificates stored in the CACs (also referred to as the "CAC root CA certificate" in this document). If you need to specify multiple CAC root CA certificates, use a comma to separate them.</p> </div> <p><b>Example:</b></p> <p><code>RootCertPath=/WEB-INF/HPCAB.cer,/WEB-INF/HPCA.cer</code></p>
CRLDownloadURL	(empty)	<p>Specifies the URL that identifies the location of your online certificate revocation list (CRL).</p> <p>You can leave this parameter empty, and then the CRL Distribution Point from the user certificate being validated is retrieved at runtime, which might cause a slightly slower performance; however, if FIPS mode is enabled in the web client (see <a href="#">"FIPS mode" on page 129</a>), you must explicitly specify this parameter (cannot leave it empty).</p> <p><b>If your root CA has multi-level intermediate CAs:</b></p>

Parameter	Default	Description
		<p>If your root CA certificate specified in <code>RootCertPath</code> has multi-level intermediate CA certificates, you should either specify the multi-level CRLs of the root CA and intermediate CA certificates (comma-separated), or leave the parameter empty.</p> <p><b>Examples:</b></p> <p><code>CRLDownloadURL=http://onsitecrl.example.com/LatestCRL.crl</code></p> <p><code>CRLDownloadURL=http://onsitecrl.example.com/LatestCRL.crl,http://onsitecrl.example.com/IntermediateCRL.crl</code></p>
<code>OCSPResponderURL</code>	(empty)	<p>Specifies the URL that identifies the location of the OCSP responder. By default, this parameter is not specified, and the OCSP responder location is retrieved from the user certificate being validated.</p> <p><b>If you specify multiple root CA certificates:</b></p> <p>If you specify multiple root CA certificates in <code>RootCertPath</code>, you must leave <code>OCSPResponderURL</code> empty. In this case, the OCSP Responder URL stored in the user certificate being validated is retrieved instead during runtime.</p> <p><b>Example:</b></p> <p><code>OCSPResponderURL=http://ocsp.example.net:80</code></p>
<code>CRLStoreLocation</code>	(empty)	<p>Specifies the location of your local CRL.</p> <div> <p><b>Note:</b> This parameter also supports multi-level intermediate CRLs (comma-separated).</p> </div> <p><b>Examples:</b></p> <p><code>CRLStoreLocation=/WEB-INF/classes/LatestCRL.crl</code></p> <p><code>CRLStoreLocation=/WEB-INF/classes/LatestCRL.crl,/WEB-INF/classes/IntermediateCRL.crl</code></p>

- c. Specify other parameters as needed.

**Note:** The validation strategy you specified determines which parameters are required, recommended or optional (see the following table).

Method	How to Enable	Parameters
Local CRL checking	validationStrategy=1	<ul style="list-style-type: none"> <li>RootCertPath (Mandatory)</li> <li>CRLStoreLocation (Mandatory)</li> </ul>
Online CRL checking	validationStrategy=2	<ul style="list-style-type: none"> <li>RootCertPath (Mandatory)</li> <li>CRLDownloadURL (Recommended): If it is not specified, the default URL stored in the user certificate is used.</li> <li>CRLRefreshScheduler (Optional): This parameter specifies an interval at which the online CRL stored in the browser cache is refreshed. The default is 1440 minutes.</li> </ul>
OCSP checking	validationStrategy=3	<ul style="list-style-type: none"> <li>RootCertPath (Mandatory)</li> <li>OCSPResponderURL (Recommended): If it is not specified, the default URL stored in the user certificate is used.</li> <li>OCSPServerCertPath (Optional): This parameter specifies the file path to the OCSP server CA certificate. If it is not specified, the RootCertPath is used as the certificate for the OCSP server.</li> </ul>
Smart Card checking	validationStrategy=4	<p>No other parameters are required.</p> <p>To ensure greater security, Smart Card checking is required for CAC cards, which are smart cards.</p>

d. Save the file.

The following is an example configuration of the cacconf.properties file.

certificateFieldExtractDN=Subject.CN

certificateOIDMapping=PrincipalName:1.3.6.1.4.1.311.20.2.3

validationStrategy=2

```
RootCertPath=/WEB-INF/CA.cer
```

```
CRLRefreshScheduler=1440
```

```
CRLDownloadURL=http://onsitecrl.verisign.com/myCompany/LatestCRL.crl
```

```
CRLStoreLocation=
```

```
OCSPResponderURL=
```

```
OCSPServerCertPath=
```

6. If a proxy is needed to access your OCSP responder URL or online CRL URL, configure a proxy in your web application server (in this example, Tomcat). You can use either of the following approaches depending on your Tomcat installation.

To configure a proxy using the Configure Tomcat shortcut:

- a. From your operating system's Start menu, click **All Programs > <Apache Tomcat> > Configure Tomcat**.
- b. On the **Java** tab, append your proxy configuration lines to the Java Options text box. For example:

```
-Dhttp.proxyHost=<Proxy Server FQDN>
```

```
-Dhttp.proxyPort=<Proxy Server Port, for example, 8080>
```

```
-Dhttp.nonProxyHosts="<Service Manager Server Host FQDN>|<Tomcat Server Host FQDN>"
```

To configure a proxy using the catalina.bat file:

- a. Open the <Tomcat>/bin/catalina.bat file in a text editor.
- b. Add the following lines:

```
set JAVA_OPTS=%JAVA_OPTS%
```

```
-Dhttp.proxyHost=<Proxy Server FQDN>
```

```
-Dhttp.proxyPort=<Proxy Server Port, for example, 8080>
```

```
-Dhttp.nonProxyHosts="<Service Manager Server Host FQDN>|<Tomcat Server Host FQDN>"
```

- c. Save the catalina.bat file.

#### Task 4. Configure each Windows client to use SSL.

When CAC sign-on is enabled for the server and web client, the Windows client must run in SSL mode.



Do the following for each Windows client:

1. Make sure that you have already configured SSL connection between the server and the Windows client. See ["Task 1. Enable required SSL encryption and trusted clients." on page 112](#)
2. Open a client connection.
3. On the **Advanced** tab, select **Use SSL Encryption**, and click **Apply**.

Task 5. Connect your web application server to the web server.

In this example, you need to connect Tomcat to Apache using the `mod_jk` module. Refer to the Apache and Tomcat documentation for detailed instructions, and in particular be sure to configure the following settings:

1. In the `httpd.conf` file, uncomment the following lines:

```
#LoadModule ssl_module modules/mod_ssl.so
#Include conf/extra/httpd-ssl.conf
#include conf/mod_jk.conf
```

2. Add the following lines in the `httpd-ssl.conf` file:

```
JkMountCopy    On
SSLOptions     +ExportCertData +StdEnvVars
```

3. Enable the following settings in the `mod_jk.conf` file, to pass the user certificate from Apache to Tomcat at runtime.

```
JkExtractSSL    On
JkHTTPSIndicator  HTTPS
JkSESSIONIndicator  SSL_SESSION_ID
JkCERTSIndicator  SSL_CLIENT_CERT
JkEnvVar    SSL_CLIENT_CERT SSL_CLIENT_CERT
JkOptions   +ForwardSSLCertChain
```

Task 6. Configure your web application server to use SSL.

**Note:** This task is required only when you do not have a web server (Apache or IIS) deployed; if your web application server is connected to a web server, skip to ["Task 7. Configure the web server to use SSL." on page 124](#)

CAC sign-on requires two-way SSL connection between the web application server and the user's browser. For information on configuring two-way SSL on your web application server, see your web application server documentation.

### Example: Configure Tomcat 7 to Use SSL

The following example assumes that you already have one root CA certificate that signs the user certificates stored in CACs and the root CA certificate is available (this root CA certificate is also referred to as the "CAC root CA certificate").

1. Generate a truststore file (for example, `truststore.jks`) for the Tomcat server.

**Caution:** Be sure to use the web tier host's fully qualified domain name (FQDN) as the Common Name (CN).

2. Import your CAC root CA certificate to the Tomcat server truststore file.

The CAC root CA certificate (`CAC_ca.crt`) is the root CA certificate that signs the user certificates stored in the CACs. Run the following command to perform the import:

```
keytool -import -v -file CAC_ca.crt -alias rootca -keystore truststore.jks -  
storepass 123456
```

3. Generate a server certificate (`server.p12`) for the Tomcat server.

- a. Generate a key pair (public and private keys).
- b. Create a certificate signing request (`.csr`) from the key pair.

```
openssl req -new -out server-req.csr -key server.key -config openssl.cnf
```

- c. Send the request to an external CA to get it signed.

**Note:** You can also create a private CA and self-sign the request.

For example, suppose you have installed and set up a Microsoft Active Directory Certificate Service (ADCS). You can send the request to this CA as described in the following:

- i. Open your ADCS URL in your browser: `http://<host>/certsrv`.
- ii. Select the **Request a certificate** task, and then click the **advanced certificate request** link.

- iii. Select the **Submit a certificate request by using a base-64-encoded CMC or PKCS #10 file...** option.
  - iv. Open the `server-req.csr` in a text editor, and copy its entire content.
  - v. Paste the request content to the Base-64-encoded certificate request box, and then click **Submit**.
  - vi. After the CA signs the request, download the signed server certificate and save it as `server.crt`.
  - vii. Download the ADCS root CA certificate, and save it as `root.crt`. Users will need to import this root CA certificate into their browser.
- d. Convert the signed server certificate to PKCS12 type (from `server.crt` to `server.p12`).
- ```
openssl pkcs12 -export -clcerts -in server.crt -inkey server.key -out server.p12
```
4. Copy the Tomcat server truststore (`truststore.jks`) and certificate (`server.p12`) to a local folder, for example, Tomcat's `/conf/ssl` folder.
  5. In the Tomcat configuration file (`/conf/server.xml`), configure SSL for CAC sign-on.
    - a. Uncomment the following lines.

```
<!--  
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"  
maxThreads="150" scheme="https" secure="true"  
clientAuth="false" sslProtocol="TLS" />  
-->
```

- b. Change `clientAuth="false"` to `clientAuth="true"`.
- c. Add the `keystoreFile` and `truststoreFile` settings.

This section now should look like the following:

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"  
maxThreads="150" scheme="https" secure="true"  
clientAuth="true" sslProtocol="TLS"
```

```
keystoreFile="../../conf/ssl/server.p12" keystorePass="changeit"  
keystoreType="PKCS12"  
  
truststoreFile="../../conf/ssl/truststore.jks" truststorePass="123456"  
truststoreType="JKS"/>
```

- d. Save your changes.

6. Restart Tomcat.

Task 7. Configure the web server to use SSL.

CAC sign-on requires two-way SSL authentication between the web server/web application server and the user's browser.

For more information on configuring two-way SSL on your web server (Apache or IIS), see your web server documentation.

#### **Example: Configure Apache 2.2 to Use SSL**

The following steps assume that you have already established a connection between Apache (the web server) and Tomcat (the web application server). See ["Task 5. Connect your web application server to the web server." on page 121](#)

1. Create a certificate and a key for the Apache server (server.crt, and server.key).
  - a. Generate a key pair (public and private keys).
  - b. Create a certificate signing request (.csr) from the key pair.

```
openssl req -new -out server-req.csr -key server.key -config openssl.cnf
```

- c. Send the request to an external CA to have it signed.

**Note:** You can also create a private CA and self-sign the request.

For example, suppose you have installed and set up a Microsoft Active Directory Certificate Service (ADCS). You can send the request to this CA as described in the following:

- i. Open your ADCS URL in your browser: `http://<host>/certsrv`.
  - ii. Select the **Request a certificate** task, and then click the **advanced certificate request** link.
  - iii. Select the **Submit a certificate request by using a base-64-encoded CMC or PKCS #10 file...** option.

- iv. Open the `server-req.csr` in a text editor, and copy its entire content.
- v. Paste the request content to the Base-64-encoded certificate request box, and then click **Submit**.
- vi. After the CA signs the request, download the signed server certificate and save it as `server.crt`.
- vii. Download the ADCS root CA certificate, and save it as `root.crt`. Users will need to import this root CA certificate into their browser.

When this step is complete, you have got the Apache server certificate (`server.crt`) and server key (`server.key`), as well as the external CA root certificate (`root.crt`). Users will need to import the `root.crt` to their browser (see ["Task 9. Configure web browsers to enable web client users to use CAC." on page 127](#)).

2. Copy your Apache `server.crt` and `server.key` to a folder. For example: `<Apache installation folder>/conf/ssl`.
3. Copy your CAC root CA certificate (`CAC_ca.crt`) to the same folder.
4. Uncomment the following line in the `httpd.conf` file.

```
#LoadModule ssl_module modules/mod_ssl.so
```

5. Configure the `\extra\httpd-ssl.conf` file as follows:

```
SSLEngine On
SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
SSLCertificateFile "<Apache installation folder>/conf/ssl/server.crt"
SSLCertificateKeyFile "<Apache installation folder>conf/ssl/server.key"
SSLCertificateChainFile "<Apache installation folder>conf/ssl/CAC_ca.crt"
SSLCACertificatePath "<Apache installation folder>/conf/ssl"
SSLCACertificateFile "<Apache installation folder>/conf/ssl/CAC_ca.crt"
SSLVerifyClient require
SSLVerifyDepth 2
```

#### Task 8. Create operator records for CAC users.

The web server extracts information from the user certificate based on the `certificateFieldExtractDN` value defined in the `<web tier installation path>\WEB-`

INF\classes\cacconf.properties file.

You must create an operator record for each CAC user. The configuration varies depending on whether Lightweight Directory Access Protocol (LDAP) is enabled on the SM server or not.

**If LDAP is not enabled for the SM server**

For each CAC user certificate, create an operator record whose Login Name (operator.name) field has the same value as the distinguished name (DN) value in the certificate.

See the following table for an example, which indicates that you need to create an operator record whose Login Name is the email address stored in the user certificate.

| In the cacconf.properties file                                                    | In the user certificate            | In the operator record              |
|-----------------------------------------------------------------------------------|------------------------------------|-------------------------------------|
| certificateFieldExtractDN=Subject.E<br><br>(DN=certificate owner's email address) | Subject.E=jack.smith@mycompany.com | Login Name=jack.smith@mycompany.com |

**If LDAP is enabled for the SM server**

Configure LDAP mapping for the operator and contacts tables, and make sure that the operator.name and contacts.operator.id fields are mapped to the same LDAP attribute that corresponds to the certificate attribute defined in certificateFieldExtractDN.

The following table lists the fields in the operator and contacts tables that you must map to an LDAP attribute.

**Note:** The LDAP attribute names here are provided only as an example, which may differ from your actual mappings. In this example, both of the name field in the operator table and the operator.id field in the contacts table are mapped to the sAMAccountName attribute, which corresponds to the user certificate attribute defined in certificateFieldExtractDN.

| File Name | Field Name   | LDAP Attribute Name |
|-----------|--------------|---------------------|
| operator  | contact.name | cn                  |
|           | full.name    | cn                  |
|           | name         | sAMAccountName      |

| File Name | Field Name   | LDAP Attribute Name |
|-----------|--------------|---------------------|
| contacts  | contact.name | cn                  |
|           | first.name   | givenName           |
|           | full.name    | cn                  |
|           | last.name    | sn                  |
|           | operator.id  | sAMAccountName      |

For details about LDAP configuration, see the *Service Manager LDAP Best Practices Guide*, available at:

**<https://softwaresupport.hp.com/group/softwaresupport/search-result/-/facetsearch/document/KM1010774>**

Task 9. Configure web browsers to enable web client users to use CAC.

Before a user can log on with a CAC, the CAC root CA certificate(s), the root CA certificate generated during your web server (or web application server) SSL configuration, as well as the user's personal certificate must be installed in the user's browser.

When users insert their CACs, the browser automatically loads their personal certificates. Users only need to manually install the root CA certificates in their browser.

As an example, the following describes how users do so in Internet Explorer.

1. Click **Tools > Internet Options > Content > Certificates**.
2. Click **Import** to import your CAC root CA certificate (CAC\_ca.crt) to the **Trusted Root Certification Authorities** certificate store.
3. Similarly, import the root CA certificate (root.crt) you generated during your web server (or web application server) SSL configuration.

Task 10. Test your CAC sign-on setup.

1. Start the Service Manager server, the web server and the web application server (in this example, Apache and Tomcat).
2. Verify that you can log in to the web client with your CAC.
  - a. Open the web tier login URL in your browser:

`http://<Your Apache Host FQDN>:<port>/<webtier>/index.do`

or if you have no Apache but only Tomcat deployed:

`http://<Your Tomcat Host FQDN>:<port>/<webtier>/index.do`

The system prompts you to select a certificate from a list of the certificates installed in your browser.

- b. Insert your CAC. The browser automatically loads your personal certificate.

**Note:** At your first CAC login, a browser refresh might be needed to make your personal certificate appear in the list.

- c. Select your personal certificate from the list.
- d. Enter your CAC PIN when prompted.

Service Manager automatically logs you in.

CAC sign-on is now successfully set up in your web tier.

3. Verify that you can log in to the Windows client through SSL encryption.
  - a. Start the Windows client, and check that the **Use SSL Encryption** option is enabled.
  - b. Enter a valid username and password to log in.

Service Manager logs you in successfully. Two-way SSL is successfully set up in your Windows client.

Now, your CAC sign-on setup is complete.



## FIPS mode

FIPS (Federal Information Processing Standards) are a set of standards that describe document processing, encryption algorithms and other information technology standards for use within U.S. non-military government agencies and by U.S. government contractors and vendors who work with the agencies.

FIPS 140-2, "Security Requirements for Cryptographic Modules," was issued by the U.S. National Institute of Standards and Technology (NIST) in May, 2001. The standard specifies the security requirements for cryptographic modules utilized within a security system that protects sensitive or valuable data.

As of version 9.32, Service Manager is FIPS 140-2 compliant when running in FIPS mode. The following table describes two operation modes of the Service Manager server and clients.

| Operation mode                                   | Description                                                                                      |
|--------------------------------------------------|--------------------------------------------------------------------------------------------------|
| FIPS mode<br>(FIPS 140-2 compliant mode)         | Supports FIPS 140-2 compliant cryptographic functions.                                           |
| Standard mode<br>(Non-FIPS 140-2 compliant mode) | Utilizes existing cryptography without the 3rd-party FIPS 140-2 validated cryptographic modules. |

To support FIPS mode, the Service Manager server and clients have introduced the following changes as of version 9.32.

**Note:** If the FIPS mode is enabled, the encrypted fields cannot be retrieved via the legacy listener.

## Server side

Out-of-the-box, the Service Manager server uses a 3rd-party FIPS 140-2 validated cryptographic module, OpenSSL FIPS Object Module. Additional changes are listed in the following table.

| Item                     | Description                                                                                                                                                                                      |
|--------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AES encryption algorithm | Service Manager provides the function to encrypt table fields. Prior to version 9.32, Service Manager uses the Data Encryption Standard (DES) encryption algorithm, which is not FIPS-compliant. |

| Item                                                   | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|--------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                        | As of version 9.32, when running in FIPS mode, Service Manager uses the Advanced Encryption Standard (AES) encryption algorithm, which is FIPS 140-2 compliant. After enabling FIPS mode, you must upgrade your database's encryption algorithm from DES to AES by running the <code>sm -upgradeencralg</code> command. For details, see <a href="#">"Configure FIPS mode in Service Manager" on the next page</a> .                                                                                                                                                                                                                                                                                                                                                                            |
| Additional support of 256-bit database encryption keys | Prior to version 9.32, Service Manager only supports 64-bit database encryption keys.<br><br>As of version 9.32, Service Manager supports 64-bit keys in non-FIPS mode and 256-bit keys in FIPS mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| The <code>fipsmode</code> parameter                    | <p>This server parameter determines whether the server runs in FIPS or non-FIPS mode when set to 1 or 0:</p> <ul style="list-style-type: none"> <li>1 (FIPS mode): All Service Manager servers and clients run in FIPS mode (for example, a pre-9.32 client, which does not support FIPS mode, can no longer connect to the SM9.32 server).</li> <li>0 (default, non-FIPS mode): All Service Manager servers and clients run in non-FIPS mode (for example, a pre-9.32 client can still connect to an SM9.32 or later server). In this mode, Service Manager keeps the same data encryption/decryption behavior as in previous versions.</li> </ul> <p><b>Note:</b> In a horizontal scaling environment, you must set this parameter to the same value (either 1 or 0) on all server nodes.</p> |
| Additional OpenSSL libraries (Windows server)          | <p>In addition to <code>libeay32.dll</code>, Service Manager has introduced <code>libeay32_fba.dll</code> and <code>libeay32_rba.dll</code>:</p> <ul style="list-style-type: none"> <li><code>libeay32_fba.dll</code>: A copy of <code>libeay32.dll</code> for backup purposes.</li> <li><code>libeay32_rba.dll</code>: A variant of <code>libeay32.dll</code>, with a relocatable base address. If the system fails to load the default dll (<code>libeay32.dll</code>), copy this file to <code>libeay32.dll</code>.</li> </ul>                                                                                                                                                                                                                                                               |

## Client side

Out-of-the-box, the Windows and web clients use a 3rd-party FIPS 140-2 validated cryptographic module, RSA BSAFE Crypto-J. To allow administrators to enable FIPS mode, the web and Windows clients have introduced the following parameters or security preference options.

| Client                                                                                       | New Parameters                                                                                        |
|----------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|
| Web (web.xml)                                                                                | <ul style="list-style-type: none"><li>• JCEProviderName</li><li>• JCEProviderClassName</li></ul>      |
| Windows ( <b>Window</b> > <b>Preferences</b> > <b>HP Service Manager</b> > <b>Security</b> ) | <ul style="list-style-type: none"><li>• JCE provider name</li><li>• JCE provider class name</li></ul> |

## Configure FIPS mode in Service Manager

**User Role:** System Administrator

Starting with version 9.32, Service Manager supports FIPS mode. To support FIPS mode, all of your server, Windows and web clients must upgrade to version 9.32 or later.

To run Service Manager in FIPS mode, you need to configure both the server and clients. By default, FIPS mode is disabled, and Service Manager uses the 64-bit DES data encryption algorithm; if you enable FIPS mode in the server, all clients must also run in FIPS mode, otherwise they cannot connect to the server.

To enable FIPS mode in a scaling environment, you must configure FIPS mode in each of the SM server nodes and clients (Windows, web, SRC, and Mobility).

This section only describes how to enable FIPS mode in the SM server and Windows and web clients. For information on how to enable FIPS mode in Service Request Catalog and the Mobility client, see the following documents:

- *Service Request Catalog Customization Guide*
- *Service Request Catalog Interactive Installation Guide*
- *Service Manager Mobile Applications User Guide*

## Server side

To enable FIPS mode in the server, follow these steps:

1. Make sure your server and applications have upgraded to version 9.32 or later.
2. Make a backup of your database.

**Caution:** FIPS mode requires the database to upgrade to the 256-bit AES encryption algorithm. Because the upgrade is irreversible, make a backup of your database.

3. Check the length of each encrypted field in your database, and increase the length if needed.

**Tip:** Use the following formula to determine a safe length for an AES encrypted field (in bytes):

$\text{Safe\_Length} = 32 + 2 * \text{source\_string\_length}$

Where: `source_string_length` is the length of the source string (for example, if you enter an 8-character password, the source string length is 8 bytes).

4. Upgrade your applications to the AES data encryption algorithm.

FIPS mode requires the use of the 256-bit AES data encryption algorithm for the database. By default, the Service Manager applications uses the 64-bit Data Encryption Standard (DES) encryption algorithm. You must upgrade the database from the DES encryption algorithm to AES by running the `sm -upgradeencralg` command.

**Caution:** This algorithm upgrade is required only when you want to enable FIPS mode.

- a. Make sure you can connect to the server successfully.
- b. Make sure all Service Manager server processes are stopped.
- c. Check that the server's configuration file (`sm.ini`) does not include the `fipsmode:1` parameter.
- d. Run the following command:

```
sm -upgradeencralg
```

**Note:** If you have many tables with encrypted fields, running this command may take quite a while; if you have already run this command on the database, running it again will have no effect and do no harm to the database.

**Tip:** You can specify a 32-character alphanumeric value for this parameter, as the new encryption key; if you do not specify any value, a default 32-character (256-bit) encryption

key is used. You can also change your encryption key later using the `changeencrkey` parameter.

- e. Check the server log (`sm.log`) for any errors.
- 5. In the server configuration file (`sm.ini`), set the following parameter. For more information, see [Startup parameter: fipsmode](#).

`fipsmode:1`

- 6. Start the server.

**Note:** If your server fails to start, and a message occurs in the server log that indicates the system failed to load `libey32.dll`, copy the `libey32_rba.dll` (in the `RUN` folder) to `libey32.dll` in the same folder, and then restart the server.

## Client side

To run the Windows and web clients in FIPS mode, you need to plug in a FIPS-certified third-party JCE provider to the Service Manager web tier and Windows client, as described in the following. Out-of-the-box, the RSA BSAFE JCE provider is used but disabled. You can enable the RSA BSAFE JCE provider or configure another JCE provider if you like.

Before you proceed, make sure you have the following information/files available. For information about any JCE providers other than RSA BSAFE, refer to the specific JCE provider documentation.

| Item                                              | Value or File(s)                                                                                                                                                                               |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| JCE provider name                                 | For RSA BSAFE:<br><code>JsafeJCE</code>                                                                                                                                                        |
| JCE provider class name                           | For RSA BSAFE:<br><code>com.rsa.jsafe.provider.JsafeJCE</code>                                                                                                                                 |
| JCE provider's jar file(s) required for FIPS mode | For RSA BSAFE, the following files are already bundled in both the SM web tier and Windows client:<br><code>cryptojcommon.jar</code><br><code>cryptojce.jar</code><br><code>jcmFIPS.jar</code> |

| Item                                             | Value or File(s)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| JCE Unlimited Strength Jurisdiction Policy Files | <p>To run in FIPS mode, both of the Windows and web clients need to update to the JCE Unlimited Strength Jurisdiction Policy Files (<code>local_policy.jar</code> and <code>US_export_policy.jar</code>), which you can download from Oracle:</p> <p><a href="http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html">http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html</a> (for JRE 8)</p> <div> <p><b>Caution:</b> The JCE Unlimited Strength Jurisdiction Policy Files are subject to import/export restrictions. HP recommends you consult the export/import control console in your country to determine if you are allowed to use these files.</p> </div> |

**Note:** The following steps use the RSA BSAFE JCE provider as an example; if you want to use another JCE provider, refer to the specific JCE provider documentation for information about the correct parameter values and jar file(s).

To configure FIPS mode in the web client, follow these steps:

1. Open the web tier configuration file (`web.xml`) in a text editor.
2. Locate the following lines:

```
<init-param>
    <param-name>JCEProviderName</param-name>
    <param-value/>
</init-param>

<!-- Specify the JCE Provider class name here, the full name includes the
package name -->
<init-param>
    <param-name>JCEProviderClassName</param-name>
    <param-value/>
</init-param>
```

3. Specify the JCE provider name and JCE provider class name.

The following are the names for RSA BSAFE.

```
<init-param>
```

```
<param-name>JCEProviderName</param-name>

<param-value>JsafeJCE</param-value>

</init-param>

<!-- Specify the JCE Provider class name here; the full name includes the
package name -->

<init-param>

    <param-name>JCEProviderClassName</param-name>

    <param-value>com.rsa.jsafe.provider.JsafeJCE</param-value>

</init-param>
```

**Caution:** Both parameter values are case-sensitive.

- 4. Copy the required jar file(s) to the web tier's WEB-INF/lib folder:

**Note:** This step is needed only for a JCE provider other than RSA BSAFE. For RSA BSAFE, the required jar files (cryptojcommon.jar, cryptojce.jar, and jcmFIPS.jar) are already in the folder.

- 5. Copy the JCE Unlimited Strength Jurisdiction Policy Files you downloaded to the web tier server's jre\lib\security directory (for example: C:\Program Files (x86) \Java\jre8\lib\security) to overwrite the two existing jar files.
- 6. (For web application servers running on Unix only) Use one of the following solutions to avoid potential performance issues.

Solution	Description
1	<p>Add a parameter -Djava.security.egd=file:/dev/./urandom to JAVA_OPTS of the web application server.</p> <p>For example, add the following line in Tomcat's catalina.sh file:</p> <p>JAVA_OPTS="\$JAVA_OPTS -Djava.security.egd=file:/dev/./urandom"</p>
2	<p>Modify the &lt;JRE_HOME&gt;/lib/java.security file used by your web application server to set securerandom.source as the following:</p> <p>securerandom.source=file:/dev/./urandom</p>

7. Restart your web application server for your configuration to take effect.
8. In your web tier's `sm.log` file, search for the following message (which indicates you have successfully configured FIPS mode):

The specified JCE provider `<jceProviderName>` has been initialized successfully.

**Note:** If you failed to configure FIPS mode, the following message occurs instead:

Initialization of the specified JCE provider `<jceProviderName>` failed.

To configure FIPS mode in the Windows client, follow these steps:

**Note:** For RSA BSAFE, skip steps 1 and 2. This is because the required modifications are already implemented for RSA BSAFE out-of-the-box.

1. Copy the JCE provider's required jar file(s) to the Windows client's third-party plugins folder:  
`<Service Manager installation path>\Client\plugins\com.hp.ov.sm.client.thirdparty_9.32.xxxx\.`
2. Update the `Manifest.mf` file in the following folder: `<Service Manager installation path>\Client\plugins\com.hp.ov.sm.client.thirdparty_9.32.0002\META-INF.`
  - a. Open the `Manifest.mf` file in a text editor.
  - b. Add the required jar file name(s) to the `Bundle-Classpath` section.

Out-of-the-box, the jar files (`cryptojcommon.jar`, `cryptojce.jar`, and `jcmFIPS.jar`) for RSA BSAFE are already included as shown below. If you want to use another JCE provider, add its required jar file names after the BSAFE ones.

```
Bundle-ClassPath: lib/,  
...  
saaj-impl-2.1.jar,  
FastInfoset-jwsdp-2.0.jar,  
cryptojcommon.jar,  
cryptojce.jar,  
jcmFIPS.jar,
```



```
<another JCE provider jar file name>,  
...
```

**Caution:** Be sure to include one leading whitespace before each jar file name.

- c. Add the JCE provider's package name to the `Export-Package` section.

For example, for RSA BSAFE the package name `com.rsa.jsafe.provider` is already included as shown below:

```
Export-Package: .,  
    COM.rsa.Intel,  
    COM.rsa.asn1,  
    COM.rsa.jsafe,  
    com.rsa.jsafe.provider,  
    ...
```

3. Configure the Windows client preferences.

- a. In the Windows client, click **Window > Preferences > HP Service Manager > Security**.

- b. Specify these parameter values (the following are values for RSA BSAFE):

- JCE provider name: `JsafeJCE`
- JCE provider class name: `com.rsa.jsafe.provider.JsafeJCE`

4. Copy the JCE Unlimited Strength Jurisdiction Policy Files you downloaded to the `<Service Manager installation path>\Client\jre\lib\security` directory to overwrite the two existing jar files.
5. Make a backup of your Windows client configuration cache, which is located in the Home folder (for example, `C:\Users\<username>\ServiceManager`), and then clear the cache.
6. Restart the Windows client for your configuration to take effect.
7. In your Windows client's `sm.log` file, search for the following message:

The specified JCE provider `<jceProviderName>` has been initialized successfully.

**Note:** If you failed to configure FIPS mode, the following message occurs instead:

Initialization of the specified JCE provider <jceProviderName> failed.

# Tokenization

As of version 9.32, the Service Manager web tier has tokenization enabled by default. Enabling tokenization helps safeguard sensitive data. When tokenization is enabled, the web tier appends a token to URLs and checks requested URLs that contain any predefined actions or match any predefined patterns to see if they have the token appended; if the token is not found, the web tier returns an error.

Administrators can also disable this feature by turning off a parameter in the web tier configuration file (`web.xml`).

## Related parameters

Three `web.xml` parameters are used for tokenization (see the following table). Out-of-the-box, the `simpleURLs` and `patterns` parameters are not present in the `web.xml` file. If you do not add them, their default values take effect.

**Note:** To enable tokenization, enabling the `antiCSRFEnabled` parameter is enough. Adding the other parameters is not necessary unless you need to change their values from the default ones.

Parameter Name	Parameter Value	Default Value	Description
<code>antiCSRFEnabled</code>	true or false	true	Set it to true to enable token validation or false to disable the feature.
<code>simpleURLs</code>	A list of comma-separated servlet actions that a URL can contain.	<code>&lt;![CDATA[service.do, detail.do, list.do, mashup.do, frames.do, printList.do, printDetail.do, uim.do, nav.menu, wf, ce, cm, /upload, /imageupload, /attachmentupload, /uniqueupload]]&gt;</code>	<p>This parameter takes effect only when <code>antiCSRFEnabled</code> is set to true. If a requested URL contains any of the listed actions and also has no token appended, the SM web tier returns a 403 error.</p> <p>When <code>antiCSRFEnabled</code> is set to false, the web tier neither checks for a token in the requested URLs, nor appends a token to any URLs.</p> <div><b>Note:</b> HP recommends</div>

Parameter Name	Parameter Value	Default Value	Description
			<p>that you use the default list. If needed, you can remove actions from the default list, but cannot add new ones.</p> <p><b>Note:</b> Token validation is skipped for the following actions: <code>index.do</code>, <code>ess.do</code>, <code>accessible.do</code>, and <code>accessible_ess.do</code>.</p>
patterns	A comma separated list of patterns	<code>&lt;![CDATA[attachments/, /download/, /colla/api/]]&gt;</code>	<p>This parameter takes effect only when <code>antiCSRFEnabled</code> is set to <code>true</code>, and is ignored when the <code>simpleURLs</code> validation has passed.</p> <p>If a requested URL matches any of the listed patterns, the web tier performs token validation on it.</p> <p><b>Note:</b> HP recommends that you use the default list. If you want, you can remove a pattern from the default list, but cannot add new ones.</p>

## Enable or disable tokenization in the web client

By default, tokenization is enabled in the web client. You can customize the tokenization settings or disable tokenization.

To customize tokenization settings:

1. In the `web.xml` file, locate the following filter and check that `antiCSRFEnabled` is set to `true`:

```
<filter>
  <filter-name>antiCSRFFilter</filter-name>
  <filter-class>com.hp.ov.web.csrf.AntiCSRFFilter</filter-class>
  <init-param>
    <param-name>antiCSRFEnabled</param-name>
    <param-value>true</param-value>
  </init-param>
</filter>
```

2. Add the other two parameters if you want to assign them a custom value.

**Note:** HP recommends that you use the default values for the two parameters; however, you are allowed to remove values from the default action/pattern list, but you cannot add new ones.

**Tip:** Ignore this step if you want to use the default values.

You should add the parameters as shown in the following.

```
<filter>
  <filter-name>antiCSRFFilter</filter-name>
  <filter-class>com.hp.ov.web.csrf.AntiCSRFFilter</filter-class>
  <init-param>
    <param-name>antiCSRFEnabled</param-name>
    <param-value>true</param-value>
  </init-param>
  <init-param>
    <param-name>simpleURLs</param-name>
    <param-value><your custom URL action list></param-value>
  </init-param>
  <init-param>
```

```
<param-name>patterns</param-name>  
  <param-value><your custom pattern list></param-value>  
</init-param>  
</filter>
```

3. Save the `web.xml` file.
4. If you want to disable tokenization, set `antiCSRFEnabled` to `false`.
5. Restart your web application server.

To disable tokenization:

1. Set `antiCSRFEnabled` to `false`.
2. Save the `web.xml` file.
3. Restart your web application server.

## Send Documentation Feedback

If you have comments about this document, you can [contact the documentation team](#) by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

**Feedback on Server security help topics for printing (Service Manager 9.41)**

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to [ovdoc-ITSM@hp.com](mailto:ovdoc-ITSM@hp.com).

We appreciate your feedback!

