HP Asset Manager 5.2

Service Catalog Integration Setup Guide

Legal Notices

Contents

Preface

Intended Audience

This document is aimed at the following personnel:

- Asset Manager Administrator

Prior knowledge of Service Manager and Asset Manager is a prerequisite to fully appreciate the contents of this document.

Typographical Conventions

`Courier` Font:

- Source code and examples of file contents.
- Commands that you enter on the screen.
- Pathnames
- Keyboard key names

Italic Text:

- Filenames, programs and parameters.
- The names of other documents referenced in this manual.

Bold Text:

- To introduce new terms and to emphasize important words.

Introduction

This document aims to provide describe how to setup Asset Manager Service Catalog integration. This document does not include configuration and troubleshooting of the system that you will find in the *Service Catalog Integration- Administration Guide*.

Architecture

The purpose of this chapter is to present the main key components of the integration, their role and also the various relationships between them.

1.1 Overview

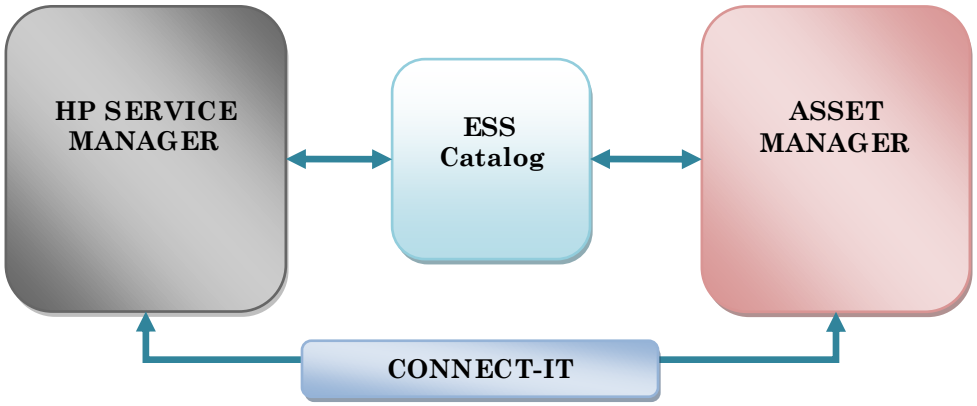


Figure 1: Global Architecture

But first, some important notions need to be defined or clarified:

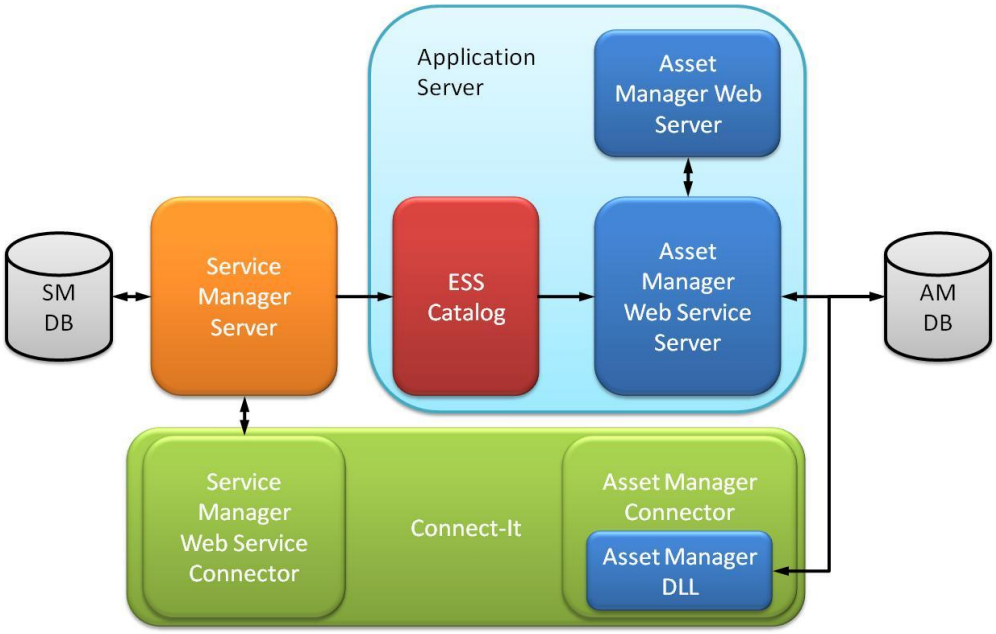
- **Asset Manager (AM)** is a complete IT management system that lets users manage a portfolio and the events associated with the lifecycle of the items in the portfolio: Procurement, cost management (tax, TCO, maintenance contracts, work orders, etc.).
- **HP Service Manager (SM)** is the application used to manage the "Employee Self Service" (ESS) Catalog. Throughout this guide, all references to HP Service Manager will always be in relationship to using this specific catalog.
- **ESS Catalog (Employee Self Service Catalog)** is an interface that is used to convert a query to create a purchase request from the HP Service Manager catalog into a series of calls to the Asset Manager Web Service which enables the purchase request to be created in Asset Manager.
- **Connect-It** is an EAI (Enterprise Application Integration) type integration platform. It is used to integrate different applications from which it can obtain or to which it can provide internal data (Internal support, equipment management software, etc.) or external data (ERP, B2B, B2C).

1.2 Required components

The components to setup are:

- Asset Manager Web Service,

- Asset Manager Web Client,
- Service Manager server,
- Connect-It,
- ESS Catalog ,
- Certificates for SSL for Service Manager (these certificates are not "components" but are required to be setup on the Service Manager server and clients machines. See Service Manager Single Sign On chapter).



All these components are not required to be on the same server. Each can be installed on a different server. The only exception is for Connect It and the Asset Manager DLL.

Single Sign On

As the integration relies both on the Asset Manager and Service Manager web interfaces, you will have to configure the overall system to use single sign on. This will ensure that users will only have to provide their login and password information to the system only once.

Single Sign On relies on the fact that the users from Asset Manager and Service Manager have the same login: the login of an account created in an authentication system. If this login is the login of the operating system of the user, the web application can be configured to not request for login/password at all. If the login is from the account of SiteMinder for example, the web application will delegate the authentication (including the form) to SiteMinder and rely on the login to identify the user.

In this document we describe how to setup SSO for Service Manager with the authentication system of the operating system and how to setup Asset Manager with SiteMinder and CAMS. You have to decide which authentication system you want to use and do the configuration accordingly.

To allow Single Sign On, Service Manager requires SSL (Secure Socket Layer) mechanism to be turned on and configured properly. SSL is a protocol to transmit private documents over the network and Internet. SSL use a cryptographic system that uses public and private keys stored in certificates

The ESS Catalog module includes the Connect-It scenario to create users in the Service Manager system for all Asset Manager users. This process is easier than creating users in both systems manually. The *Service Catalog Integration - Administration Guide* provides more details about the profile of the users and how to use the Connect-It scenario. In this guide, the purpose of setting up single sign on is to achieve a connection to the system, using one login (the login of the account created in the Asset Manager authentication system (IIS, Active Directory, SiteMinder, etc.), and presenting access to Asset Manager and Service Manager's user interfaces.

1.3 Asset Manager Single Sign On

This document describes how to setup single sign on for Asset Manager using the SiteMinder or CAMS as authentication systems. If you decide to use another authentication system, please do the configuration accordingly.

1.3.1 Site Minder integration

1.3.1.1 Technical Design

SiteMinder provides user authentication and authorization When a web application (such as AC Web) is protected by SSO, the user is redirected to an authentication server where they are presented with a logon page. Once the user logs on, the authentication server will verify that the user has access to the web application. It the user does have access, the authentication server will redirect the user back to the initial web application. In addition to the redirect, the authentication server will append information about the user within the HTTP header data. This header data can then be used as needed by the web application.

Although the SSO authentication server can authenticate and authorize a user for a particular web application, AssetCenter has its own unique logon process. To utilize SSO as the logon method requires the following:

- A working Single Sign-on tool (such as SiteMinder) with established accounts and access to the AM Web application. (The process to protect a particular web application will vary depending on the tool. Please see your SSO administrator for information on what is required.)
- Creation of a new JavaBean that AM Web will use to pull HTTP header information supplied by SSO. This will be used by AC Web to allow a user to automatically log in.
- Modification of AC Web configuration files to use this sign on process.

Employee's username within AC (amEmplDept.UserLogin) must match the username used within SSO.

1.3.1.2 Requirements, Guidelines, and Considerations

- Experience with AssetCenter, AssetCenter Web
- Familiarity with web development and related technologies (i.e. Tomcat)
- Java development experience (will require creating and compiling custom Java classes)

1.3.1.3 Workflow and Tasks

This section describes the tasks involved for setting up SSO with AC 5.0 Web.

Step Action/Process Action Description Required Input

Setup SSO to protect the AssetCenter Web URL

SSO must be configured to protect the AssetCenter Web. Contact the SSO administration team.

Establish SSO accounts for users that will require access to AssetCenter Web.

All users that will access AssetCenter Web will require SSO accounts. These accounts must have usernames that match the employee's username (login) within AssetCenter. Verify that the SSO accounts exist and that the username of the SSO account matches the value stored in the employees AC account (value is stored in amEmplDept.UserLogin for each user).

Create SSO authentication JavaBean

Develop custom JavaBean that will extract HTTP header information passed from SSO. The JavaBean should pull the validated username and return that value to the AC Web logon process. The requirements for the JavaBean will depend on how the HTTP header information is passed from SSO. See section 5 below for examples. Once the JavaBean has been developed, it will need to be compiled via the Java SDK. To compile successfully, the classpath must include several references to the AssetCenter Web APIs. See section 5 for examples.

Verify that JavaBean compiled successfully with no errors.

Add the newly created JavaBean class to the AssetCenter Web applicationcontext.xml configuration file.

Open and edit the TOMCAT\webapps\AssetCenter\WEB-INF\classes\applicationcontext.xml file. Add the name of the new JavaBean to the section below (found in the acwc:filterChainProxy):

Example:

Before:

```
/**=acwc:ACSesssionSerializationFilter,acwc:httpSessionContextIntegrationFilter,acwc:preAu
thenticationFilter,acwc:authenticationProcessingFilter,acwc:anonymousProcessingFilter,acwc
:ACPostAndInternalUrlFilter,acwc:filterSecurityInterceptor
```

After:

```
/**=acwc:ACSesssionSerializationFilter,acwc:httpSessionContextIntegrationFilter,acwc:SSOAu
thenticationFilter,acwc:preAuthenticationFilter,acwc:authenticationProcessingFilter,acwc:a
nonymousProcessingFilter,acwc:ACPostAndInternalUrlFilter,acwc:filterSecurityInterceptor
```

Add the new JavaBean configuration information under the acwc:preAuthenticationFilter section:

```
<bean id="acwc:SSOAuthenticationFilter"
class="com.hp.ov.ac.web.security.AcSSOAuthenticationFilter">
  <property name="authenticationManager">
    <ref bean="acwc:authenticationManager"/>
  </property>
  <property name="defaultRole">
    <value>ROLE_PRE</value>
  </property>
  <property name="keepDomain">
    <value>true</value>
  </property>
</bean>
```

Modify AssetCenter Web applicationcontext.xml configuration file to handle SSO with nonblank user passwords.

(This change allows SSO to work for accounts where the AssetCenter user login has an existing password. Without this change, the password for each user account within AC must be blank.) Open and edit the TOMCAT\webapps\AssetCenter \WEB-INF\classes\applicationcontext.xml file. In the configuration section of the acwc:jaasAuthenticationProvider bean, add the following line:

```
<bean class="com.hp.ov.cwc.security.jaas.AuthenticationDetailsCallbackHandler"/>
```

Example:

Before:

```
<bean class="org.acegisecurity.providers.jaas.JaasNameCallbackHandler"/>
<bean class="org.acegisecurity.providers.jaas.JaasPasswordCallbackHandler"/>
```

After:

```
<bean class="org.acegisecurity.providers.jaas.JaasNameCallbackHandler"/>
<bean class="org.acegisecurity.providers.jaas.JaasPasswordCallbackHandler"/>
<bean class="com.hp.ov.cwc.security.jaas.AuthenticationDetailsCallbackHandler"/>
```

Restart Tomcat after all changes have been made.

1.3.1.4 JavaBean Examples

JavaBean Requirements

During the log on process, AssetCenter will call the newly created JavaBean and verify the user was authenticated by an SSO process. This is done via a call to a method named getAuthenticatedUsername. The JavaBean must implement this method returning the username of the authenticated user (from the HTTP header). If the HTTP header value does not exist (the user entered the logon page outside of the SSO process) the JavaBean should return NULL.

For the authentication process to work, the username passed from the getAuthenticatedUsername method must match the username stored in the employee table (amEmplDept) of the AssetCenter database.

1.3.1.5 SiteMinder Example

This example is based on an implementation where SiteMinder authentication using Tomcat on a Windows Server environment is not possible. To work around this issue, an IIS website must be developed. This website would be protected by SiteMinder. Authentication would occur as follows: ASP or ASPX page would take the HTTP header information passed by SiteMinder and HTTP POST the header to AssetCenter: JavaBean would use POST data (rather than HTTP header) for authentication.

Example ASP page

```
<%@ LANGUAGE = VBScript %>
<HTML>
<BODY>
<FORM name="f" action="http://SERVERNAME:PORTNUMBER/AssetCenter/"
method="post">
<INPUT type="hidden" name="AUTH_USER"
value="<%=Request.ServerVariables( "HTTP_NTUSERDOMAINID" )%>" />
</FORM>
</BODY>
</HTML>
<SCRIPT LANGUAGE="JavaScript">
self.document.forms[0].submit()
</SCRIPT>
Example JavaBean:
package com.hp.ov.ac.web.security;
import com.hp.ov.cwc.security.acegi.PreAuthenticationFilter;
import javax.servlet.http.HttpServletRequest;
public class AcSSOAuthenticationFilter extends PreAuthenticationFilter
{
  public AcSSOAuthenticationFilter()
  {}
  protected String getAuthenticatedUsername(HttpServletRequest httpservletrequest)
  {
    String s="";
    if(httpservletrequest.getParameter( "AUTH_USER" )!=null){
      s = httpservletrequest.getParameter( "AUTH_USER").replace(':','\\');
    }
    if(s!="")
      return s;
    return null;
  }
}
```

Compiling JavaBeans

To compile the examples above, the classpath to several Tomcat and AssetCenter libraries must be referenced.

Example

```
JSDKHOME\bin\javac.exe -classpath C:\TOMCAT_HOME\common\lib\servletapi.
jar;C:\TOMCAT_HOME\webapps\AssetCenter\WEB-INF\lib\security-3.02-
SNAPSHOT.jar;C:\TOMCAT_HOME\webapps\AssetCenter\WEB-INF\lib\spring-
1.2.7.jar; AcSSOAuthenticationFilter.java
```

1.3.2 CAMS Integration

1.3.2.1 Software Prerequisites

Before starting, you must have the following items:

· A working SSO platform.

· A working installation of HP AssetCenter Web 5.x.

· Installation package of the web server you want to use. Typical choices are IIS and Apache.

· Installation package of the module you want to use to handle the communication between your web server and your web application server (mod_jk, isapi filter).

1.3.2.2 Task Prerequisites

1. Install Apache 2.0.X.
2. Configure it to use port 81.
3. Install the connector mod_jk. To find the correct version for your OS, see

http://www.apache.org/dist/tomcat/tomcat-connectors/jk/binaries/

1.3.2.3 Apache Configuration

1. Rename the downloaded library to mod_jk.so.
2. Copy it to your (Apache home)/modules directory.
3. Create a workers.properties file
4. Create a mod_jk.log for mod_jk specific logs.
5. Add the URL sections that you will let mod_jk handle.

```
LoadModule jk_module modules/mod_jk.so
JkWorkersFile "C:/Program Files/Apache Software Foundation/Tomcat
5.0/conf/jk/workers.properties"
JkLogFile "C:/Program Files/Apache Software Foundation/Tomcat
5.0/conf/jk/mod_jk.log"
JkLogLevel emerg
JkMount /AssetCenter/* ajp13
JkMount /jsp-examples/* ajp13
# Define 1 real worker using ajp13
worker.list=ajp13
# Set properties for the ajp13 worker
worker.ajp13.type=ajp13
worker.ajp13.host=localhost
worker.ajp13.port=8009
worker.ajp13.lbfactor=50
worker.ajp13.cachesize=10
worker.ajp13.cache_timeout=600
worker.ajp13.socket_keepalive=1
worker.ajp13.recycle_timeout=300
```

1.3.2.4 CAMS Policy Server and Agent

The following example lists the settings you can use to quickly set up a test platform. In this example, CAMS will use its own user credential repository for controlling authentication. For additional information about installation and set-up, see the CAMS documentation.

Policy Server

Install the policy server according to the information provided by the installation documentation. In the following example, the "system" domain is used by CAMS by default. All prerequisites are built into the policy server tree structure. After verifying that the installation has been successful, perform the following configuration steps.

1. Edit \conf\domains\system\login-config.xml and set up the Apache agent log-in script.

Replace:

```
<login-parameters>
  <login-parameter name="camsLoginUrl" value="/cams/login.jsp"/>
</login-parameters>
```

With:

```
<login-parameters>
  <login-parameter name="camsLoginUrl" value="/cgi-bin/camslogin.
pl"/>
</login-parameters>
```

2. Edit \conf\domains\system\cams-users.xml and add an Admin user that is similar to the

AssetCenter Admin user. Assign the Admin user the default initial password of "password."

```
<user name="Admin" password="password" roles="administrator"/>
```

Apache Agent

Before proceeding, you should install Perl on your computer. There is a free implementation of Perl for Windows. Go to http://www.activestate.com/Products/ActivePerl/ for more information.

1. After you have installed Perl, follow the agent installation instructions and choose the following folder as your installation folder: C:/cams-webagent-apache2

2. Edit the apache httpd.conf file by adding:

```
LoadModule CamsApache20WinntWebAgent_module "C:/cams-webagentapache2/
cams/mod_cams_apache20_winnt_webagent.so"
CamsWebAgentHome "C:/cams-webagent-apache2"
```

3. Copy the files from the C:\cams-webagent-apache2\cgi-bin folder to the Apache cgi-bin directory.

1.4 Service Manager Single Sign On

You can configure the HP Service Manager Client 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 log-on screen and directly enter the application. Since users only have to enter logon information once, trusted sign-on is also known as single sign-on.

1.4.1 SSL and Single sign-on

Single sign-on relies on a working SSL with Client-Authentication configuration, and integration with a trusted authentication source such as Integrated Windows Authentication or a network security management tool.

1.4.2 Creation of SSL Certificates

Any SSL certificate has to be issued by a trusted source, known as the **Certificate Authority (CA)**. You can purchase the server and client certificates from a registered certificate authority (CA), such as VeriSign™, or you can generate your self-signed certificates for the Certificate Authority and create both the server and client certificates. This document assumes you are choosing the second option.

Tools for creating the certificates are: OpenSSL and keytool executables. OpenSSL is used to create the private key and certificate for your private CA and sign your server and client certificates. Keytool is used to create the server and client keystore files, generate the Certificate Signing Request and import/export the certificate to a keystore file. To run OpenSSL a configuration file is required.

1.4.2.1 Sample openssl.conf file

```
[ req ]
default_bits              = 2048
default_keyfile           = privkey.pem
distinguished_name        = req_distinguished_name
attributes                = req_attributes
x509_extensions           = v3_ca

dirstring_type = nobmp

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = US
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State
stateOrProvinceName_default     = CA

localityName                    = Locality Name (eg, city)
localityName_default            = San Diego

organizationName                = Organizational Name
organizationName_default        = HPSW

organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default  = BTO

commonName                      = Common Name (eg, computer hostname)
commonName_max                  = 64
commonName_default              = server.domain.com
emailAddress                    = Email Address
```

```
emailAddress_max                = 40
emailAddress_default            = user@domain.com

[ req_attributes ]
challengePassword               = A challenge password
challengePassword_min           = 4
challengePassword_max           = 20

[ v3_ca ]

subjectKeyIdentifier            = hash
authorityKeyIdentifier          = keyid:always,issuer:always
basicConstraints                = CA:true
```

Two Windows batch files are used, tso_srv_svlt.bat and tso_cln_svlt.bat. The tso_srv_svlt.bat batch file is used to generate the self-signed CA certificate and the server certificate. The tso_cln_svlt.bat batch file is used to generate the client certificate and to append the client's public key to SM's "trusted client keystore file". Run the batch file from a DOS command window in the directory that contains the OpenSSL executables or add the path to the directory containing the OpenSSL executables to your environment variables.

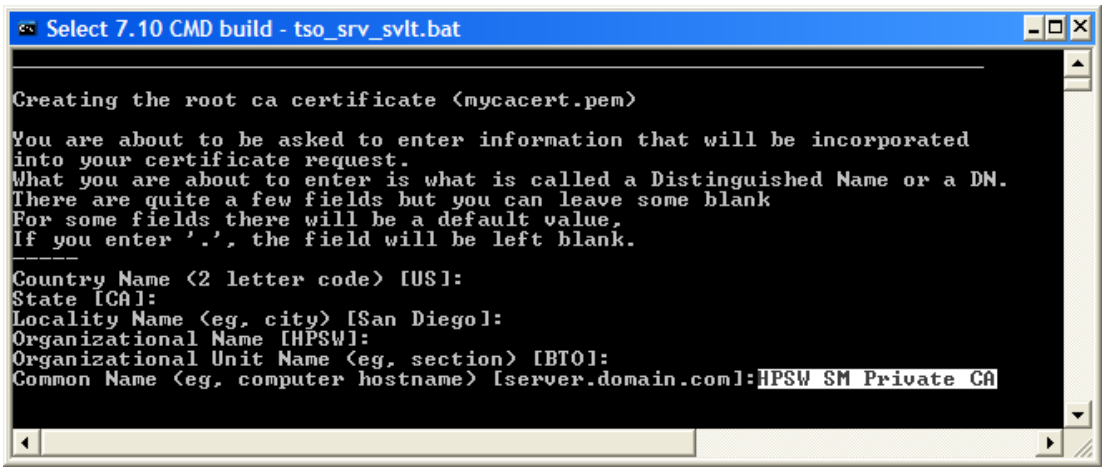
To run tso_srv_svlt.bat just type the batch file name in the DOS command window. To create the client certificate, type tso_cln_svlt client.host.fully.qualified.domain.name. After you run the batch file, three directories are created under your working directory:

certs - contains all the certificate files
crs - contains all the certificate signing request files
key- contains all the key files

Copying certificates:

Created Files	Copy To:
certs\cacerts	<SM_Server_installation_path>\RUN directory
	<web_application_server_installation_path>\WEB-INF directory
certs**trustedclients.keystore**	<SM_Server_installation_path>\RUN directory
key\ **client.host.fully.qualified.domain .name.keystore**	*<SM Client>*/plugins/com.hp.ov.sm.client.common_7.xx directory
	<web_application_server_installation_path>\WEB-INF directory
key**server.keystore**	<SM_Server_installation_path>\RUN directory

- ***Note: The "Common Name" for your root ca certificate doesn't have to be the host.fully.quaified.domain.name. For example, it could be "HPSW SM Private CA".***



1.4.2.2 tso_srv_svlt.bat

```
REM #
REM # SM SSL Certificates Creator (server component)
REM #
REM # This batch file facilitates the creation of the SSL certificates
REM # that are needed to setup SSL encryption for Service Manager 7.0x.
REM #
REM # Run this batch file only once to create the certificates for the REM # Service
Manager server.
REM #
REM #-----------------------------------------------------------------
cls

@echo off

SETLOCAL enableextensions

REM # Openssl settings
REM #
REM # This batch file uses the openssl.conf file as input for the
REM # openssl program. All _default values can be set according to your
REM # organisation.
REM #
REM # Only one openssl.conf is needed.
REM #
REM #-----------------------------------------------------------------
set OPENSSL=openssl

REM # Java Settings
REM #
REM # set the JAVA_HOME variable to the installation path of the JRE you
REM # want to use.
REM #
REM #-----------------------------------------------------------------
if not "%JAVA_HOME%" == "" goto gotJavaHome
set JAVA_HOME="C:\Program Files\Java\jre1.5.0_13"

:gotJavaHome
set JAVA_HOME=%JAVA_HOME%\jre
if exist "%JAVA_HOME%\bin\keytool.exe" goto okJava
echo The JAVA_HOME environment variable is not defined correctly
goto end

:okJava
PATH=%JAVA_HOME%;.
echo PATH=%PATH%
echo %JAVA_HOME%
```

```
set KEYTOOL=%JAVA_HOME%\bin\keytool

REM # Password settings
REM #
REM # These are the default password settings used by the openssl and
REM # keytool programs. All passwords can be changed, EXCEPT the
REM # CACERT_PASSWD, as this is the default password that the SUN cacert
REM # from the JRE uses!
REM #
REM #-----------------------------------------------------------------
set CAROOT_PASSWD=caroot
set CACERT_PASSWD=changeit
set SERVER_KEYSTORE_PASSWD=serverkeystore
set CLIENT_KEYSTORE_PASSWD=clientkeystore
set TRUSTEDCLIENTS_KEYSTORE_PASSWD=trustedclients

@del /q key
@del /q certs
@del /q crs

@mkdir key
@mkdir certs
@mkdir crs

copy %JAVA_HOME%\lib\security\cacerts %JAVA_HOME%\lib\security\cacerts.orig
copy %JAVA_HOME%\lib\security\cacerts certs\cacerts

REM #-----------------------------------------------------------------
REM # Private Key & Root Certificate generation
REM #-----------------------------------------------------------------

REM create the private key for your private CA
@echo.
@echo ____________________________________________________________________
@echo.
@echo Creating a Self-Signed Certificate (cakey.pem)
@echo.
%OPENSSL% genrsa -des3 -passout pass:%CAROOT_PASSWD% -out key/cakey.pem 2048
@echo.
@echo ____________________________________________________________________
@echo.

REM create the root CA cert
@echo.
@echo ____________________________________________________________________
@echo.
@echo Creating the root ca certificate (mycacert.pem)
@echo.
%OPENSSL% req -new -key key/cakey.pem -x509 -days 1095 -out certs\mycacert.pem -config
./openssl.conf -passin pass:%CAROOT_PASSWD%
@echo.
@echo ____________________________________________________________________
@echo.

REM import the certificate into the System-wide keystore
@echo.
@echo ____________________________________________________________________
@echo.
@echo Importing the certificate into the System-wide keystore (cacerts)
@echo.
%KEYTOOL% -import -keystore certs/cacerts -trustcacerts -alias servicemanager -file
certs/mycacert.pem -storepass %CACERT_PASSWD%
@echo.
@echo ____________________________________________________________________
@echo.

copy certs\cacerts %JAVA_HOME%\lib\security



REM #-----------------------------------------------------------------
```

```
REM # Server Key & Certificate generation
REM #-----------------------------------------------------------------

REM generate private server key and keystore
@echo.
@echo ____________________________________________________________________________
@echo.
@echo Creating the Server keystore (server.keystore)
@echo.
%KEYTOOL% -genkey -alias smserver -keystore key/server.keystore -storepass
%SERVER_KEYSTORE_PASSWD%
@echo.
@echo ____________________________________________________________________________
@echo.

REM generate the server request certificate to be signed using our CA key & cert
@echo.
@echo ____________________________________________________________________________
@echo.
@echo Generating the Server request certificate (servercert_request.crs)
@echo.
%KEYTOOL% -certreq -alias smserver -keystore key/server.keystore -file
crs/servercert_request.crs -storepass %SERVER_KEYSTORE_PASSWD%
@echo.
@echo ____________________________________________________________________________
@echo.

REM sign the server request certificate using our CA
@echo.
@echo ____________________________________________________________________________
@echo.
@echo Signing the Server request certificate (smservercert.pem)
@echo.
%OPENSSL% x509 -req -days 1095 -in crs/servercert_request.crs -CA certs/mycacert.pem -
CAkey key/cakey.pem -CAcreateserial -out certs/smservercert.pem -passin
pass:%CAROOT_PASSWD%
@echo.
@echo ____________________________________________________________________________
@echo.

REM import the server certificate into the keystore
@echo.
@echo ____________________________________________________________________________
@echo.
@echo Importing Server certificate into Server keystore
@echo.
%KEYTOOL% -import -trustcacerts -alias smserver -keystore key/server.keystore -file
certs/smservercert.pem -storepass %SERVER_KEYSTORE_PASSWD%
@echo.
@echo ____________________________________________________________________________
@echo.

copy %JAVA_HOME%\lib\security\cacerts.orig %JAVA_HOME%\lib\security\cacerts
del %JAVA_HOME%\lib\security\cacerts.orig

:end
ENDLOCAL disableextensions
```

1.4.2.3 tso_cln_svlt.bat

```
REM #
REM # SC-SM SSL Certificates Creator (client component)
REM #
REM # This batch file facilitates the creation of the SSL certificates that are needed to
setup SSL encryption for Service Manager 7.0x.
REM #
REM # Run this batch file with the fully-qualified domain name of the client machine as
the first argument (%1), from the command line :
REM #
```

```
REM # \prompt>tso_cln_svlt <fully-qualified domain name>
REM #
REM # Rerun this batch file for each client machine to create a unique
REM # set of certificates for the Service Manager Eclipse or Web client.
REM #
REM #-----------------------------------------------------------------
cls

@echo off

SETLOCAL enableextensions

REM # Openssl settings
REM #
REM # This batch file uses the openssl.conf file as input for the openssl program. All
_default values can be set according to your
REM # organisation.
REM #-----------------------------------------------------------------
set OPENSSL=openssl


REM # Java Settings
REM #
REM # set the JAVA_HOME variable to the installation path of the JRE you
REM # want to use.
REM #
REM #-----------------------------------------------------------------
if not "%JAVA_HOME%" == "" goto gotJavaHome
set JAVA_HOME="C:\Program Files\Java\jre1.5.0_13"

:gotJavaHome
set JAVA_HOME=%JAVA_HOME%\jre
if exist "%JAVA_HOME%\bin\keytool.exe" goto okJava
echo The JAVA_HOME environment variable is not defined correctly
goto end

:okJava
PATH=%JAVA_HOME%;.
echo PATH=%PATH%
echo %JAVA_HOME%

set KEYTOOL=%JAVA_HOME%\bin\keytool

REM # Password settings
REM #
REM # These are the default password settings used by the openssl and
REM # keytool programs. All passwords can be changed, EXCEPT the
REM # CACERT_PASSWD, as this is the default password that the SUN
REM # cacert from the JRE uses..!!
REM #
REM #-----------------------------------------------------------------
set CAROOT_PASSWD=caroot
set CACERT_PASSWD=changeit
set SERVER_KEYSTORE_PASSWD=serverkeystore
set CLIENT_KEYSTORE_PASSWD=clientkeystore
set TRUSTEDCLIENTS_KEYSTORE_PASSWD=trustedclients

if exist "key\cakey.pem" goto okcakey
echo You need to create a self-signed CA certificate prior this process
goto end

:okcakey
```

```
copy %JAVA_HOME%\lib\security\cacerts %JAVA_HOME%\lib\security\cacerts.origcopy
copy certs\cacerts %JAVA_HOME%\lib\security\cacerts

echo Client Key and Certificate creation

REM #-----------------------------------------------------------------
REM # Client Key & Certificate generation
REM #-----------------------------------------------------------------

REM generate private client key and keystore
@echo.
@echo ___________________________________________________________________________
@echo.
@echo Creating the Client keystore (%1.keystore)
@echo.
%KEYTOOL% -genkey -alias %1 -keystore key/%1.keystore -storepass %CLIENT_KEYSTORE_PASSWD%
@echo.
@echo ___________________________________________________________________________
@echo.

REM generate the Client request certificate to be signed using our CA key
REM & cert
@echo.
@echo ___________________________________________________________________________
@echo.
@echo Generating the Client request certificate (clientcert_request.crs)
@echo.
%KEYTOOL% -certreq -alias %1 -keystore key/%1.keystore -file crs/clientcert_request.crs -
storepass %CLIENT_KEYSTORE_PASSWD%
@echo.
@echo ___________________________________________________________________________
@echo.

REM sign the Client certificate using our CA
@echo.
@echo ------------------------------------
@echo.
@echo Signing the Client request certificate (scclientcert.pem)
@echo.
%OPENSSL% x509 -req -days 1095 -in crs/clientcert_request.crs -CA certs/mycacert.pem -
CAkey key/cakey.pem -CAcreateserial -out certs/scclientcert.pem -passin
pass:%CAROOT_PASSWD%
@echo.
@echo ___________________________________________________________________________
@echo.

REM import the client certificate into the keystore
@echo.
@echo ___________________________________________________________________________
@echo.
@echo Importing Client certificate into Client keystore
@echo.
%KEYTOOL% -import -trustcacerts -alias %1 -keystore key/%1.keystore -file
certs/scclientcert.pem -storepass %CLIENT_KEYSTORE_PASSWD%
@echo.
@echo ___________________________________________________________________________
@echo.

REM #-----------------------------------------------------------------
REM # Adding the client Certificate to Trusted Keystore
REM #-----------------------------------------------------------------
```

```
REM export client public key/certificate
@echo.
@echo ____________________________________________________________________
@echo.
@echo Exporting Client public certificate from Client keystore (clientpubkey.cert)
@echo.
%KEYTOOL% -export -alias %1 -keystore key/%1.keystore -file certs/clientpubkey.cert -
storepass %CLIENT_KEYSTORE_PASSWD%
@echo.
@echo ____________________________________________________________________
@echo.

REM import public key/certificate into the keystore
@echo.
@echo ____________________________________________________________________
@echo.
@echo Importing Client public certificate into Trustedclients keystore
(trustedclients.keystore)
@echo.
%KEYTOOL% -import -alias %1 -file certs/clientpubkey.cert -keystore
certs/trustedclients.keystore -storepass %TRUSTEDCLIENTS_KEYSTORE_PASSWD%
@echo.
@echo ____________________________________________________________________
@echo.


copy %JAVA_HOME%\lib\security\cacerts.origcopy %JAVA_HOME%\lib\security\cacerts
del %JAVA_HOME%\lib\security\cacerts.origcopy

:end
ENDLOCAL disableextensions
```

1.4.3 SSL Configuration for SM

It's a good practice to verify that the certificates that you just created are working on the "SSL ONLY" configurations before you start configuring SSO. The "SSL ONLY" configurations are: SSL Server-Side Authentication and SSL with Client-Authentication.

1.4.3.1 SSL Server-Side Authentication

SSL Server-Side Authentication is used to test that the server certificate was created correctly. To enable the SSL server:

- Add the following parameters to the sm.ini file and restart the SM server for the changes to take effect.
  # parameters for starting SSL connector
  sslConnector: 1
  ssl: 1
  httpPort: HPPT port number
  httpsPort: HPPTS port number

  # keystore for trusted certificate authority (CA) certificates

truststoreFile: The keystore file contains the certificate authority's certificate
ex: cacerts
truststorePass: The password for the cacerts
ex: changeit

SM server's keystore file and password
keystoreFile: The keystore file contains the SM server's certificate and private key
ex: server.keystore
keystorePass: The password for SM server's keystore file
ex: serverkeystore

The following lines should appear in the sm.log file:

```
  1732(  2468) 09/12/2008 08:48:36 JRTE I Initializing Service Manager servlet "SM 7 Servlet"
  1732(  2468) 09/12/2008 08:48:36 JRTE I Initializing Service Manager servlet "SM Servlet"
  1732(  2468) 09/12/2008 08:48:37 JRTE I Initializing Service Manager servlet "SM 7 Servlet"
  1732(  2468) 09/12/2008 08:48:37 JRTE I Initializing Service Manager servlet "SM Servlet"
  1732(  2468) 09/12/2008 08:48:37 Initializing Coyote HTTP/1.1 on http-13080
  1732(  2468) 09/12/2008 08:48:37 Starting Coyote HTTP/1.1 on http-13080
  1732(  2468) 09/12/2008 08:48:38 Initializing Coyote HTTP/1.1 on http-13081
  1732(  2468) 09/12/2008 08:48:38 Starting Coyote HTTP/1.1 on http-13081
  1732(  2468) 09/12/2008 08:48:38 JRTE I Started Tomcat - HTTP port is 13080
  1732(  2468) 09/12/2008 08:48:38 JRTE I Started Tomcat - HTTPS port is 13081
```

- No client configuration is required.
  Start the SM Windows client and connect to the SM server.

  "SSL connection accepted" should appear in the sm.log file:

```
  1732(  2544) 09/12/2008 08:48:43  RTE I Thread B9C4DE327CB77DE9B264713A6B1F0857
initialization done.
  1732(  2544) 09/12/2008 08:48:43  RTE D Parsing request document: <?xml version="1.0"
encoding="utf-
........
  1732(  2544) 09/12/2008 08:48:44 JRTE I SSL connection accepted
```

1.4.3.2 SSL Client-Authentication

SSL with Client-Authentication is used to verify that the client certificate was created correctly. To enable Client-Authentication:

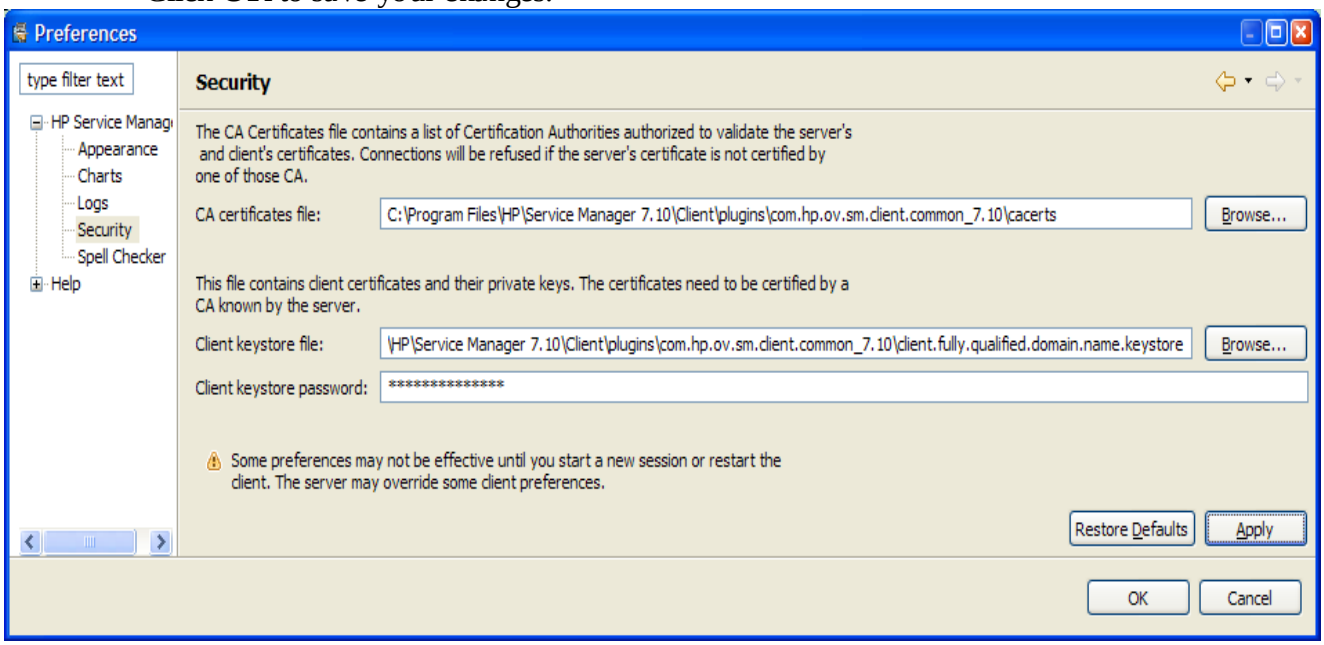
- Add the following parameters to the modified sm.ini file - the one in which SSL Server-Side Authentication has been enabled - and restart the SM server for the changes to take effect.

  ssl_reqClientAuth:1

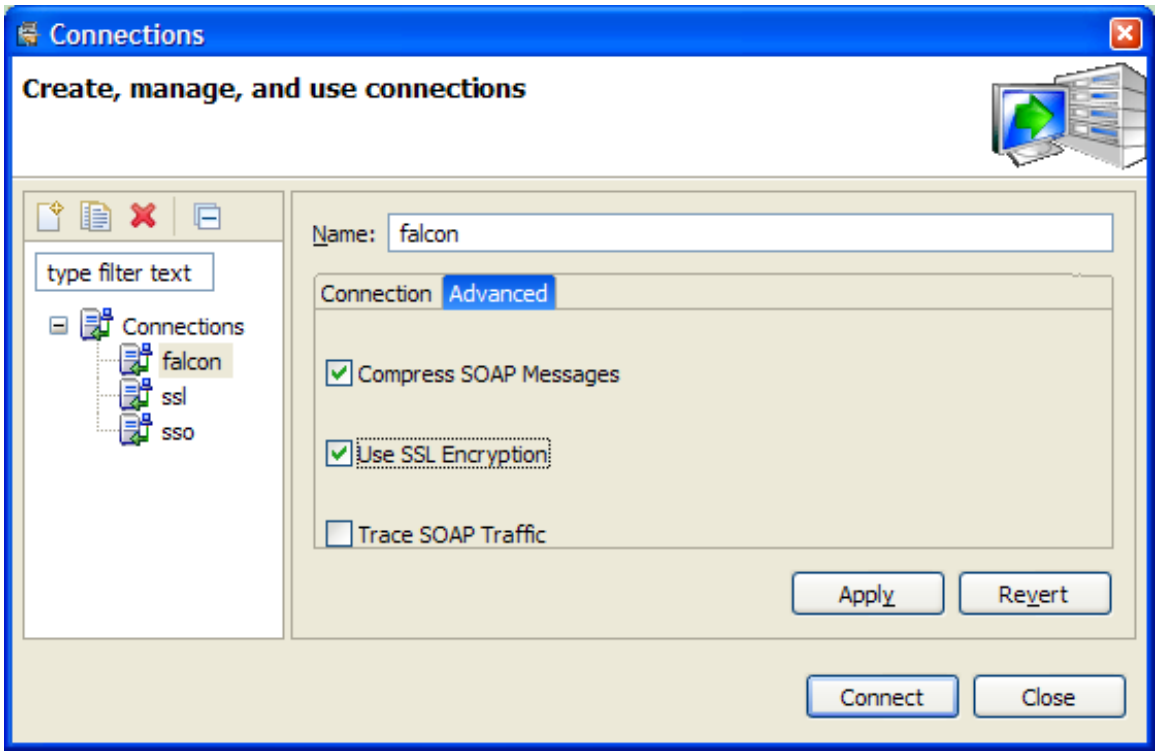
- Configure the SM Windows client to use SSL encryption.

  Open the HP Service Manager Windows client.

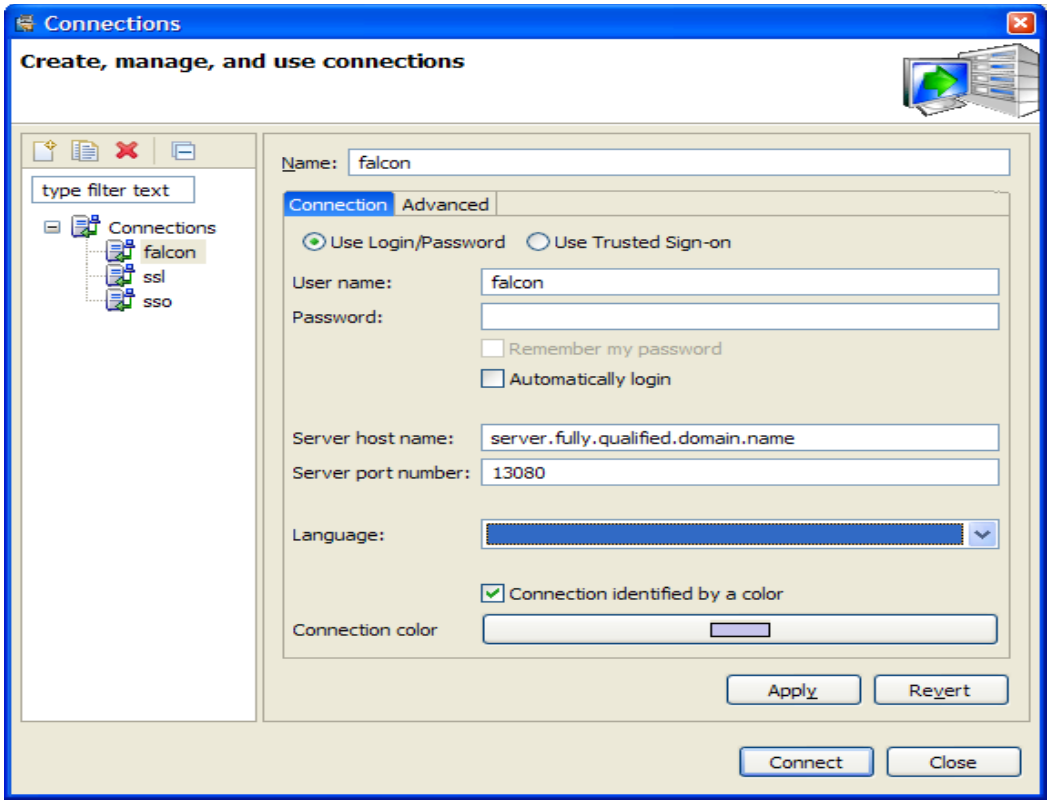
- Setup Security Preferences
  Click **Window** > **Preferences** > **HP Service Manager** > **Security**.
  Click **Browse** for the **CA certificates file** field.
  Browse to the path to the cacerts keystore.
  Click **OK** to accept the path.
  Click **OK** to save your changes.



- Create a client connection with SSL turned on
  Click **File > Connect** -> **Connections** to open the Service Manager Connections window.
  Click the **Advanced** tab and click to select the **Use SSL Encryption** option.



Click the **Connection** tab and, in the Server host name field, enter the "fully qualified domain name".
Click **Apply** -> **Close** to apply and save the options.



Now, you restart the SM server and the SM Windows client for the changes to take effect. In the sm.log file, the line shown in italics below confirms that the SSL setup was successful:

```
7792(  7992) 09/12/2008 12:50:05  RTE I Thread 52C0D5F57349820777681DAF93BA45A5 initialization
done.
7792(  7992) 09/12/2008 12:50:05  RTE D Parsing request document: <?xml version="1.0"
encoding="utf-
........
7792(  7992) 09/12/2008 12:50:06 JRTE I SSL connection accepted
```

- Configure SSL for Web Tier.
  Stop the Web application server running the SM Web Tier.
  Open the Web configuration file, web.xml, in a text editor.
  - ❖ Make sure that the serverHost parameter contains the fully qualified name of the Service Manager Server:
    <init-param>
    <param-name>serverHost</param-name>
    <param-value>***servername.domainname.com***</param-value>
    </init-param>

  - ❖ Turn on SSL encryption:
    <init-param>
    <param-name>ssl</param-name>
    <param-value>**true**</param-value>
    </init-param>
  - ❖ Set the cacerts parameter to the keystore file that contains your server's certificate authority.
    <init-param>
    <param-name>cacerts</param-name>
    <param-value>/WEB-INF/cacerts</param-value>
    </init-param>
  - ❖ Specify the client's private keystore to use in encrypted communication.
    <init-param>
    <param-name>keystore</param-name>
    <param-value>>/WEB-
INF/**client.host.fully.qualified.domain.name.keystore**</param-value>
    </init-param>
  - ❖ Specify the password for the client's private keystore
    <init-param>
    <param-name> keystorePassword</param-name>
    <param-value>**enter keystore password here**</param-value>
    </init-param>

  Restart the Web application server. If you are using Tomcat then you should see the following lines in Tomcat's stdout_<date>.log file:
  Sep 14, 2008 08:17:40 PDT [INFO] Found SSL client keystore: /WEB-INF/clientcerts
  Sep 14, 2008 08:17:40 PDT [INFO] Found SSL CA certificate keystore: /WEB-INF/cacerts
  Start a Web browser and a SM Web client session.

In the sm.log file, the line shown in italics below confirms that the SSL setup was successful:

```
7792(  2544) 09/12/2008 13:05:05  RTE I Thread B9C4DE327CB77DE9B264713A6B1F0857 initialization
done.
7792(  2544) 09/12/2008 13:05:05  RTE D Parsing request document: <?xml version="1.0"
encoding="utf-
........
7792(  2544) 09/12/13:50:05  JRTE I SSL connection accepted
```

1.4.4 Single Sign-on Configuration

In the previous steps you have created and checked the private CA, server and client certificates. The SSL configuration is now complete and operational. At this stage, you may now setup SSO for SM.

- Add the following parameters to the modified sm.ini file - the in which SSL with Client-Authentication has been enabled - and restart the SM server for the changes to take effect.

  # Single sign-on capability is enabled
  trustedsignon:1

  # Clients are required to present signed certificates to the server
  # and need to be on the list of trusted clients
  ssl_reqClientAuth:2

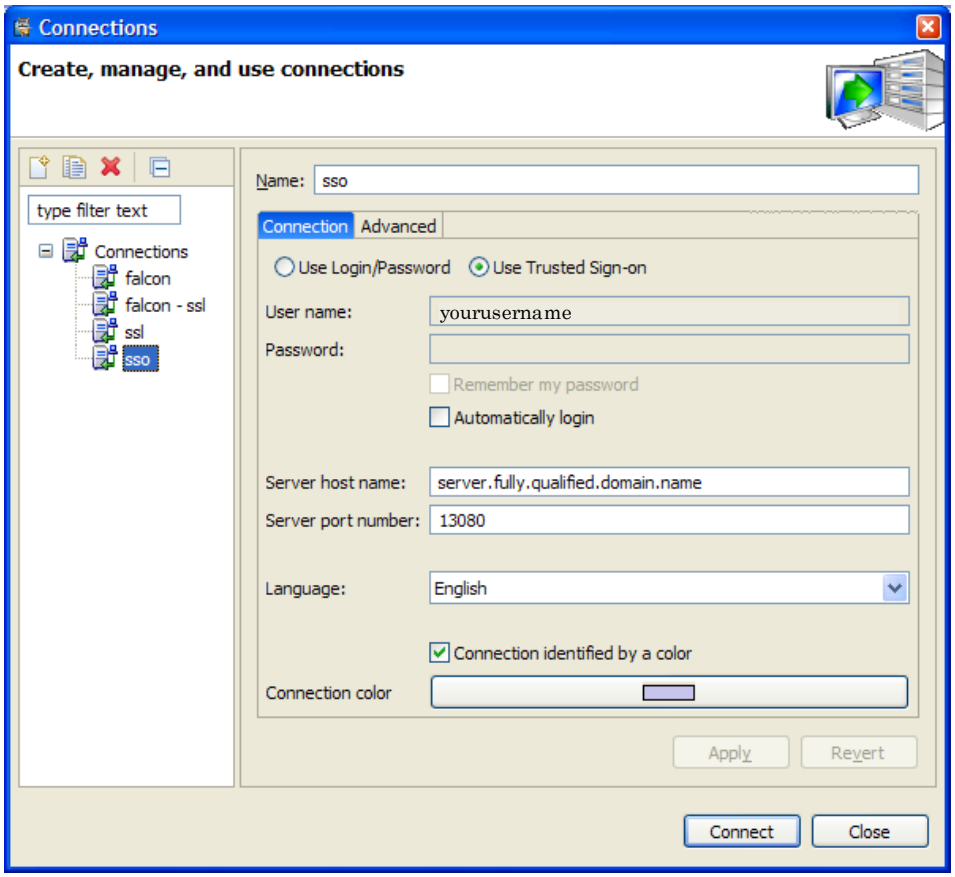
  # keystore file containing the signed certificates of trusted SM clients
  ssl_trustedClientsJKS: trustedclients.keystore

  # Password to the trusted client keystore
  ssl_trustedClientsPwd: trustedclients

- Test the SSO Configuration with the SM Windows client
  To use trusted sign-on, ensure that you have an operator record with the same username and password as those you use to log on to the network.

  Open the SM Windows client; the one where the SSL configuration has been tested.

  - Enable Trusted Sign-on
    Click **File > Connect** -> **Connections** to open the Service Manager Connections window.
    Click the **Connection** tab
    Click "Use Trusted Sign-on"
    Click Connect



In the sm.log file, the line shown in italics below confirms that the SSO setup was successful:

```
5552(  2252) 09/12/2008 15:13:42  RTE I Thread DB2E22E2A0E0C641E94D565094D5BA1C initialization
done.
  5552(  2252) 09/12/2008 15:13:42  RTE I SOAP client information scguiwswt 7.1.005
(DAILY.240) at 15.80.161.30
  5552(  2252) 09/12/2008 15:13:42 JRTE I SSL connection accepted
  5552(  2252) 09/12/2008 15:13:42  RTE I Set trusted sign-on login user to yourusername
  5552(  2252) 09/12/2008 15:13:42  RTE I User yourusername has logged in and is using a Named
license (1 out of a maximum 25)
```

- Configuring the SM Web client to use SSO
  1. In order to use trusted sign on, set the value of the isCustomAuthenticationUsed parameter to false. This enables Service Manager to send the current user name in the HTTP header. In the <**path to Web application**>**/WEB-INF** /web.xml
     <context-param>
     <param-name>isCustomAuthenticationUsed</param-name>
     <param-value>false</param-value>
     </context-param>
  2. In the <**path to Web application**>**/WEB-INF/classes/**application-context.xml file, replace
     /**=httpSessionContextIntegrationFilter,anonymousProcessingFilter
     With
     /=httpSessionContextIntegrationFilter,**preAuthenticationFilter**,

anonymousProcessingFilter
Save your changes and restart the Web server on the Service Manager Web Tier.

- Configuring the Web server and Web application server
  Note: The following steps assume that the Web server and Web application server configurations have already been implemented, and that the only required changes to the configuration of those servers are those described in this document.

  ❖ Tomcat configuration changes

  1. If you are using Tomcat 5.0.x, enter
     request.tomcatAuthentication=false
     at the end of the /tomcat/conf/jk2.properties file.
  2. Starting in Tomcat 5.5.x, jk2.properties is no longer used by default. For Tomcat 5.5.x, include the tomcatAuthentication=”false” parameter in the jk2 worker port definition.
  3. Open the Tomcat/conf/server.xml file in the same directory and search for the following line:

     <!-- Define a Coyote/JK2 AJP 1.3 Connector on port 8009 -->
     Change the parameters in this section from:
     <Connector port=”8009”
     enableLookups="false" redirectPort="8443" debug="0" protocol="AJP/1.3" />
     To:
     <Connector port=”8009”
     enableLookups="false" **tomcatAuthentication="false"** redirectPort="8443"
     debug="0" protocol="AJP/1.3" />
  2. Save the file.
  3. Restart Tomcat for the changes to take effect.

  ❖ 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 http.conf file to allow for trusted sign-on:
     #SspiAuth Module
     LoadModule sspi_auth_module modules/mod_auth_sspi.so

     2.1. For mod_auth_sspi.so module prior version 1.3
     <Location "/sm710">
     AllowOverride None
     Options None
     Order allow,deny
     Allow from all

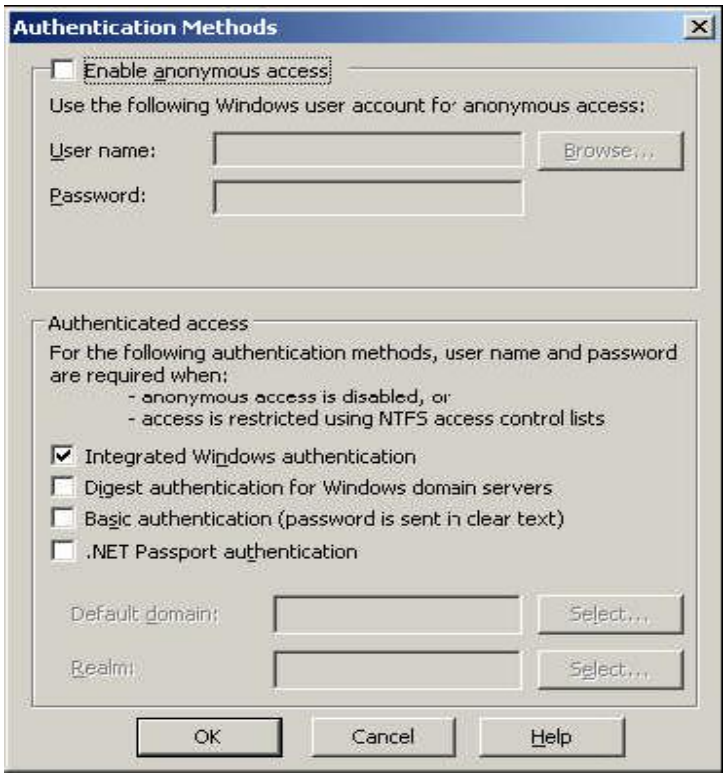
```
    SSPIAuth On
    SSPIDomain AMERICAS
    SSPIAuthoritative On
    SSPIOfferBasic Off
    require valid-user

  </Location>
```

2.2. For mod_auth_sspi.so module version 1.3+

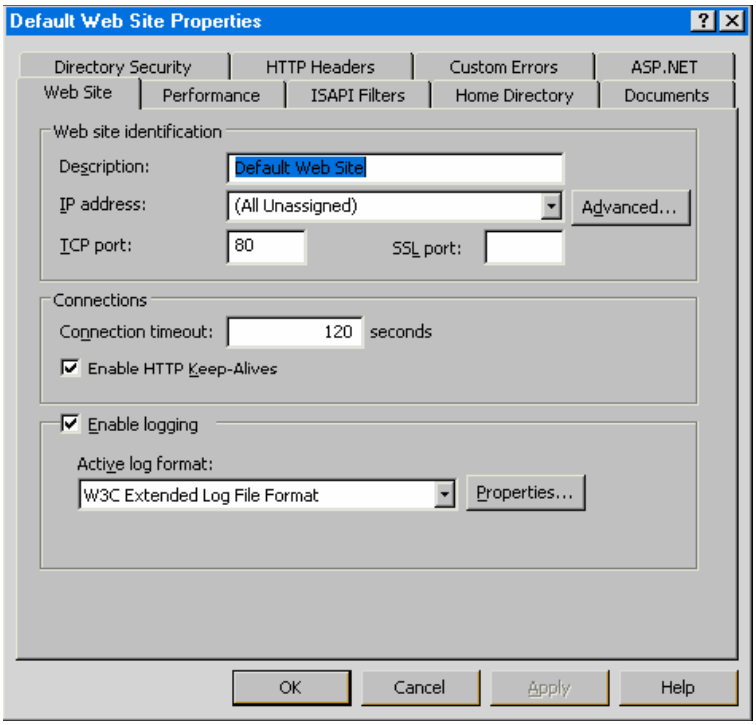
```
<Location "/sm">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
    AuthType SSPI
    SSPIAuth On
    SSPIDomain MYDOMAIN
    SSPIAuthoritative On
    SSPIOfferBasic Off
    SSPIPerRequestAuth On
    require valid-user
</Location>
```

The name within the Location tag needs to be the path the user enters to open the Service Manager Web Client Web site. This is usually /sm, since the name is taken from the sm.war file. For configurations with multiple domains, comment out the SSPIDomain parameter by adding a crosshatch character (#) in front of the line.

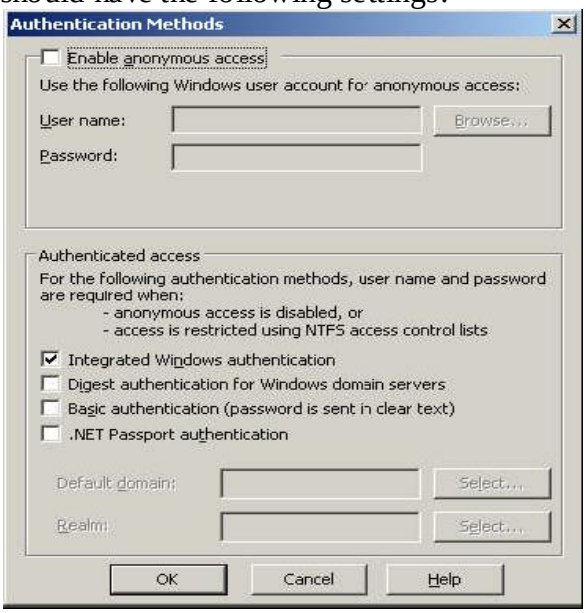
- Internet Information Server
  In the Properties window for the Service Manager virtual directory, click the Directory Security tab and enter the information as shown below:



- Configuring Internet Information Server version 6
  1. Open the IIS Manager (Start – Administrative Tools – Internet Information Services (IIS) Manager)
  2. Click on Web Service Extensions
  3. Set extension status to Allowed for All Unknown ISAPI Extensions
  4. Optionally set Active Server Pages to Allowed.
  5. Check the properties of the Default Web Site.



6. Go to the ISAPI Filters tab and check that the green upward arrow in the Status column is pointing up.
7. Go to the Directory Security tab and click on the Edit button in the Authentication and access control frame. The Authentication Methods page should have the following settings:



Make sure you disable "Enable anonymous access" and enable "Integrated Windows authentication"
Optionally, you may enable Advanced Digest Authentication.

Additional Information: Advanced Digest Authentication is an extension of Digest security. Digest security uses MD5 hashing to encrypt user credentials (user name, password and user roles).

Basic authentication sends the user name and password details over the network in base64 encoded format. These details can be easily "sniffed" (captured with a protocol analyzer) and decoded by an intruder, who could then use the credentials for nefarious purposes. Digest security's MD5 hash enhances security by applying cipher algorithms that are more sophisticated and more difficult to crack. An MD5 hash is binary data consisting of the encrypted user name, password and realm. The 'realm' is the name of the domain that authenticates the user.

The MD5 hash is embedded into an HTTP 1.1 header thus is only supported by HTTP 1.1-enabled browsers. Digest or Advanced Digest authentication mechanisms can not be enabled if the target browsers do not support HTTP 1.1.

Advanced Digest Security takes the Digest authentication model a bit further by storing the user credentials on a domain controller as an MD5 hash in the Active Directory database. Intruders would need to get access to the Active Directory to steal the credentials. This adds another layer of security to protect access to Windows 2003 Web sites.

Both Digest and Advanced Digest Authentication only work on Web Distributed Authoring and Versioning (WebDAV) enabled directories. WebDAV (formerly called Web Folders) is a secure file transfer protocol that lets people download, upload, and manage files on remote computers across the internet and intranets WebDAV is similar to the File Transfer Protocol (FTP) except that WebDAV always uses password security and data encryption on file transfers, whereas FTP doesn't support those features.

When you enable this feature, you'll get the message: "Digest authentication only works with Active Directory domain accounts. For more Information about configuring Active Directory domain accounts to allow digest authentication click Help. Are you sure you want to continue (Yes, No, Help). Clicking on Help gives the following information:

Digest Authentication Warning

The authenticated access method, Digest authentication, applies only to domain accounts on servers running Microsoft® Windows® Server 2003 and requires the accounts to store passwords using reversible encryption. Internet Information Services (IIS) sends a hash value rather than the password over the network, working across proxy servers and other firewalls.

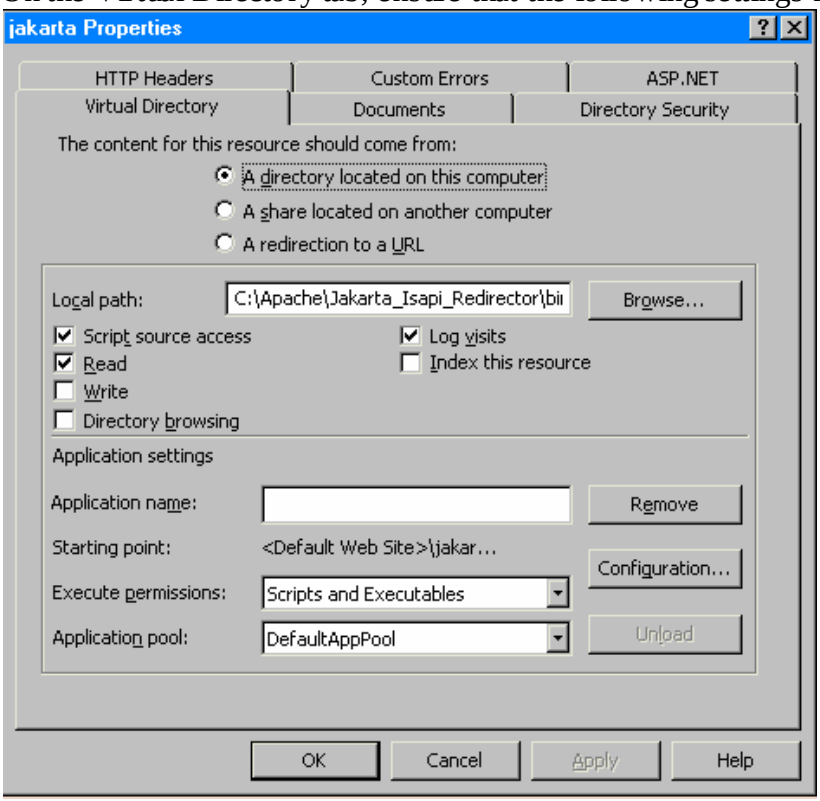
Requirements for Digest Authentication

Before enabling Digest authentication on your server running IIS, ensure that all of the following minimum requirements are met. Only domain administrators can verify that the domain controller requirements are met. Check with your domain administrator if you are unsure about whether your domain controller meets the following requirements:

- *All clients that access a resource that is secured with Digest authentication are using Microsoft Internet Explorer 5.0 or later.*
- *The user and the server running IIS must be members of, or be trusted by, the same domain.*
- *Users must have a valid Windows user account stored in Active Directory® on the domain controller.*
- *The domain must have a Windows 2000 or later domain controller.*
- *The IIS server must be running a member of the Windows Server 2003 family or later.*

8. In the Default Web Site Folder, right click on Jakarta and select Properties

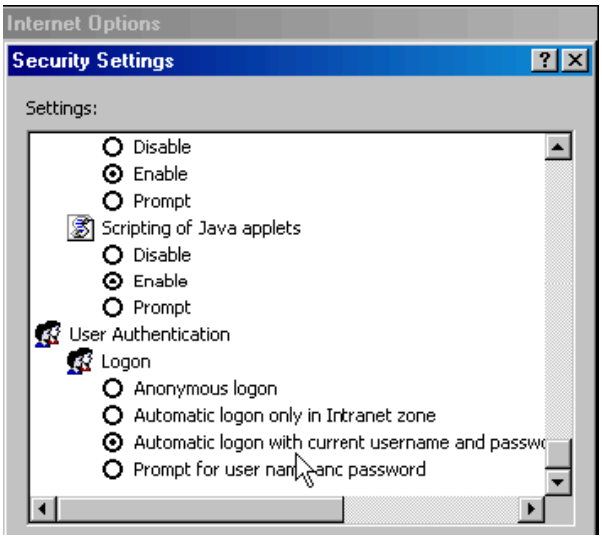
On the Virtual Directory tab, ensure that the following settings have been set:



9. On the Directory Security Tab, click on Edit in the Authentication and access control frame.
10. Ensure that Enable anonymous access is disabled and Integrated Windows authentication is enabled.
11. Restart the Internet Information Server service.

❖ Internet Explorer security settings

Open the Internet Explorer Options Security Settings dialog. In the User Authentication Logon section, enable the Automatic logon with current username and password setting:



1.4.5 Troubleshooting

Error message:

keytool error: Failed to establish chain from reply

Cause:

This message is issued during the import of a certificate when the cacerts file in the <JAVA_HOME>/lib/security folder is not the same as the cacerts file used to create the certificates. To fix this issue, copy the cacerts file used when building the certificates to the <JAVA_HOME>/lib/security folder.

Tip:

- Make sure your Private CA's Certificate is in the cacerts file; run keytool:
  NOTE: The –alias should point to your Private CA's alias name.

```
C:\Temp\SM SSO Docs\certs>keytool -list -v -alias servicemanager -keystore cacerts
Enter keystore password:  changeit
Alias name: servicemanager
Creation date: Sep 10, 2008
Entry type: trustedCertEntry
Owner: EMAILADDRESS=user@domain.com, CN=HP Support Private CA, OU=BTO, O=HPSW, L=San
Diego, ST=CA, C=US
Issuer: EMAILADDRESS=user@domain.com, CN=HP Support Private CA, OU=BTO, O=HPSW, L=San
Diego, ST=CA, C=US
Serial number: 8a383be82792b2eb
Valid from: Wed Sep 10 21:06:04 PDT 2008 until: Sat Sep 10 21:06:04 PDT 2011
Certificate fingerprints:
        MD5:  31:6D:37:E5:4A:F2:DE:E9:19:C9:34:4F:8A:ED:AF:37
        SHA1: A9:03:2A:D4:65:10:B6:9D:68:FF:35:7A:15:B8:54:9A:B5:DA:B6:DA
```

- Make sure your server and client certificates were signed by your Private CA. Below is a sample listing of the server certificate.

```
C:\Temp\SM SSO Docs\key>keytool -list -v -keystore server.keystore
Enter keystore password:  serverkeystore
Keystore type: jks
Keystore provider: SUN
Your keystore contains 1 entry
Alias name: smserver
Creation date: Sep 10, 2008
Entry type: keyEntry
Certificate chain length: 2
Certificate[1]:
Owner: CN=swsh013.rose.hp.com, OU=HPSW IT, O=HP, L=SD, ST=CA, C=US
Issuer: EMAILADDRESS=user@domain.com, CN=HP Support Private CA, OU=BTO, O=HPSW, L=San Diego, ST=CA,
C=US
Serial number: a767f29753b08dcd
Valid from: Wed Sep 10 21:08:46 PDT 2008 until: Sat Sep 10 21:08:46 PDT 2011
Certificate fingerprints:
        MD5:  43:C5:B0:60:D8:D3:44:F7:5A:2F:AD:06:B0:F1:C1:44
        SHA1: 34:DC:A3:56:1B:33:C0:8D:2B:BC:40:51:D6:26:E9:96:7F:98:65:6C
Certificate[2]:
Owner: EMAILADDRESS=user@domain.com, CN=HP Support Private CA, OU=BTO, O=HPSW, L=San Diego,
ST=CA, C=US
Issuer: EMAILADDRESS=user@domain.com, CN=HP Support Private CA, OU=BTO, O=HPSW, L=San Diego, ST=CA,
C=US
Serial number: 8a383be82792b2eb
Valid from: Wed Sep 10 21:06:04 PDT 2008 until: Sat Sep 10 21:06:04 PDT 2011
Certificate fingerprints:
        MD5:  31:6D:37:E5:4A:F2:DE:E9:19:C9:34:4F:8A:ED:AF:37
        SHA1: A9:03:2A:D4:65:10:B6:9D:68:FF:35:7A:15:B8:54:9A:B5:DA:B6:DA
*******************************************
*******************************************
```

Error Message:

Not a trusted client. IP/host name: <*IP Address of Client*>/<*Hostname of Client*>

Cause:

The hostname of the client that sent the request was different from the DN in the client's certificate. To fix this, recreate the client certificate correctly.

Tip:

Restart the SM server with –debughttp option to see more error messages.

Error Message:

No SSL certificate was presented by the peer!

Cause:

The request was an HTTPS request, but no client certificate is available. Ensure that the web.xml or the windows client preferences point to the correct client certificate.

Error Message:

SSL debug: Could not load trusted client file.

Cause:

Service Manager could not find the trusted client JKS file to which the ssl_trustedClientsJKS parameter points. Verify that the parameter points to the correct location.

Error Message:

Client <*DN in the client's certificate*> is not in the trusted list file.

Cause:

The client's certificate is not in the trusted list file.
To fix this:

export the client’s public certificate and then import it to the trusted client store

keytool -export -alias *clientalias* -keystore *clientkeystore* -file *clientpubkey.crt*
keytool -import -alias *clientalias* -file *clientpubkey.crt* -keystore *trustedclients.keystore*

Validating the system

To validate the setup you should be able to go through this process:

- Open an Internet browser,
- Go to Asset Manager Web url,
- You should not be prompted for a login/password combination if you've configured Asset Manager single sign-on to use the operating system's login information. If you choose another authentication provider, just log in using the user and password from the authentication system.
- From the same browser, open the Service Manager Web url
- You should not be prompted for a login/password combination.

Support

HP Software support Web site

You can visit the HP Software Support web site at:

www.hp.com/go/hpsoftwaresupport

This Web site provides a list of contacts and information about products, services and support provided by HP Software.

HP Software online software support provides users with self-healing services to help them resolve their problems. It also provides a quick and efficient means to access interactive technical support tools to manage specific issues. As a technical support customer, you can use the support site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP Software 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 sign in. Many also require a valid support contract. To find more information about support access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

To register for an **HP Passport ID**, go to:

http://h20229.www2.hp.com/passport-registration.html

Limited responsibility clause

Asset Manager is integrated with several third-party applications.

Examples: Database engines, Web servers, single sign-on software, load-balancing and clustering hardware and software solutions, reporting software such as Crystal Reports, etc.

Support for these applications is limited to their interface with Asset Manager.

Support does not cover installation problems, setup and customization problems nor malfunctioning of the third-party application.