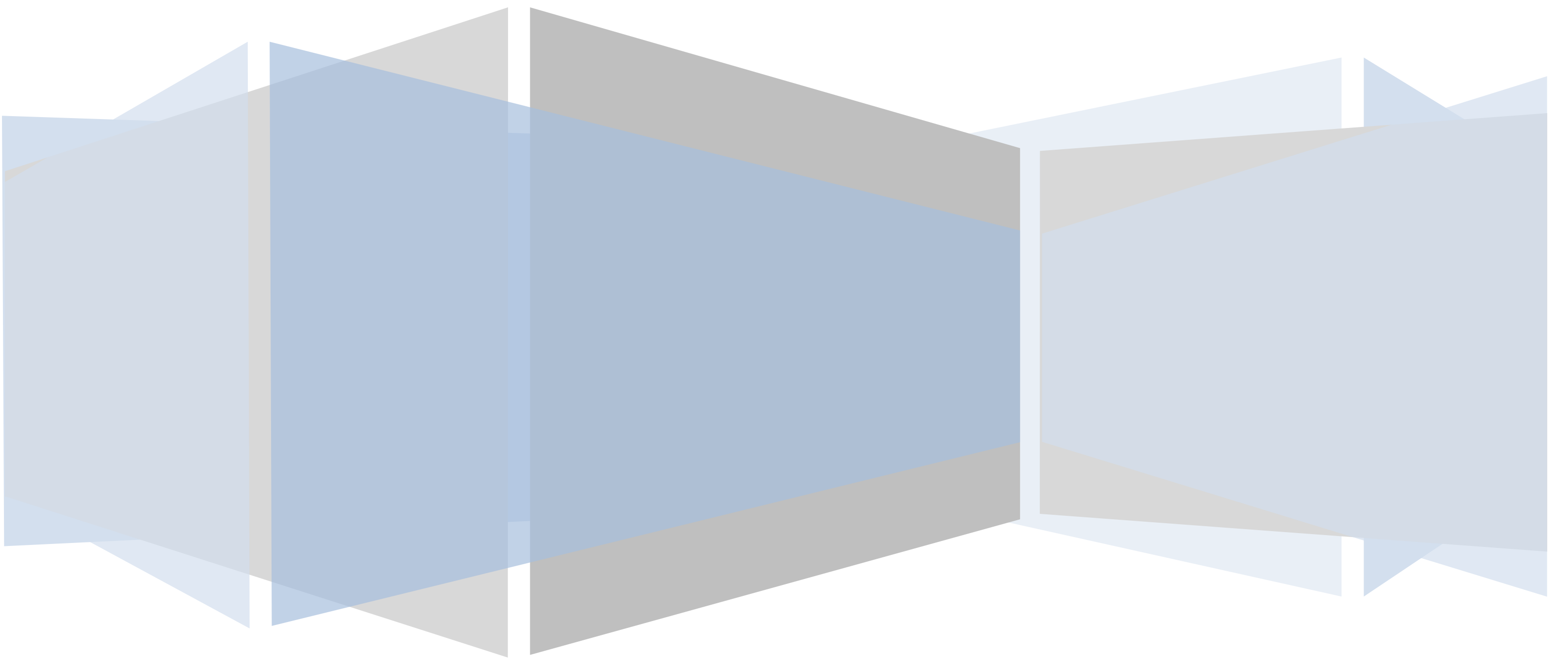
SSO Plugin

Authentication service for HP, Kinetic, Jasper, SAP and CA products

J System Solutions

http://www.javasystemsolutions.com

Version 4.0



Introduction

This guide provides SSO implements to a range of products using the SSO Plugin Authentication Service. This is a standalone edition of SSO Plugin that can be added to third party applications and in conjunction with a little configuration, brings all of the SSO Plugin integrations (Active Directory, AD, X509, etc) to the third party application.

The JSS support website contains all the SSO Plugin documentation and videos covering installation and configuration.

Implementing SSO

There are a number of steps to this process and they are split into three categories:

1. Copying files to the target web application (see below).
2. Configuring the target web application by modifying files such as the application's web.xml file.
3. Configuring SSO Plugin, of which more information can be found in the Configuring SSO Plugin document (called configuring-midtier-webtier.pdf) - this is currently bias towards BMC Mid Tier and HP Web Tier, but the SSO configuration instructions relevant.

Prior to performing the steps, stop the Java web server (ie Tomcat) instance running the application. When the steps have been completed, start it again.

Licensing

The product is licensed by generating a license in the support area of the JSS website.

The licensing tool allows two month trial licenses to be generated, or if you've purchased the product, a permanent license can be generated by entering the hostname of the server running SSO Plugin.

After generating a license, enter it into the SSO Plugin 'Configuration' page.

Copying the SSO Plugin files to the target application

Locate the authentication-service directory within the SSO Plugin Authentication Service download package.

Copy the contents to the target web application, ie tomcat/webapps/application, where application is src, Asset Manager, jasperserver, etc. Copying will ask you to confirm the folder replace because WEB-INF already exists. Please select ***yes***.

No files will be replaced, only added to the application.

Next, proceed to configure the application following the instructions in the relevant section below.

Patching the web.xml file

In order for SSO Plugin to intercept requests to the application and perform single-sign on, the application's web.xml file requires patching.

This can be performed automatically by SSO Plugin or the file can be manually patched.

Start by locating the file relevant patch file in the patches directory within the installation files.

Automatic patching

Copy the patch file into the web application WEB-INF directory and rename it web.xml.patch.

Automatic patching will occur when SSO Plugin is configured through the web based user interface.

Manual patching

Open the patch file and copy the contents to the clipboard.

Locate the application web.xml file within the WEB-INF directory. Make a backup of the file.

Open the web.xml file in your favourite text editor, locate the last </context-param> or </description> element and paste the patch immediately after it.

If in doubt, send the application web.xml file to JSS for assistance.

Configuring HP Service Request Catalog 1.3+

Start by copying the SSO Plugin files to the target web application.

Deployment approaches

We provide two approaches for deploying SSO Plugin to SRC. This approach is suitable if you do not require user aliasing and do not wish to use built-in Active Directory authentication, which due to a bug in the Adobe Flash browser plugin, causes the product to fail when Internet Explorer negotiates with NTLM.

In the event your requirements include user aliasing or AD integration, consult the installation for Service Request Catalog document for the alternative strategy.

Patching application files

The application web.xml file must be patched by following the instructions in the section patching the web.xml file. The relevant patch file is web.xml.patch.src.

applicationContext.properties

Locate the applicationContext.properties file, found in the SRC program directory, ie tomcat/webapps/src/WEB-INF/classes/applicationContext.properties. The required changes to enable SSO are different for SRC versions 1.3 and 1.4.

SRC 1.3

Locate the following:

```
# Security Mode: Choose your security method
```

And uncomment to enable the following option, commenting whatever is currently enabled:

```
src.security.mode=remoteUsrSsoUiAndTsoWs
```

SRC 1.4/9.34+

Locate the src.security.mode variable and set to tso:

```
src.security.mode=tso
```

Next, locate the src.security.ssoEnabled variable and set it to true:

```
src.security.ssoEnabled=true
```

Modifying the Spring security configuration

SRC 1.3/1.4

The Spring security file is located in the SRC web application WEB-INF/spring/security directory. The filename is different in SRC 1.3 and 1.4 but the same change is made for both versions.

Locate the file for the relevant version:

- SRC1.3: applicationContext-security-remoteUsrSsoUiAndTsoWs.xml.
- SRC1.4: applicationContext-security.xml.

Open the file in a text editor, find the line containing authenticationProcessingFilterEntryPoint and add the text in bold below this line:

```
<http entry-point-ref="authenticationProcessingFilterEntryPoint">
  <intercept-url pattern="/jss-sso/**" filters="none" />
  <intercept-url pattern="/logout.jsp" filters="none" />
  <intercept-url pattern="/secure/nosso.jsp" filters="none" />
```

SRC 9.34+

Locate the applicationContext-security.xml file, open the file in a text editor, find the line containing authenticationProcessingFilterEntryPoint and add the text in bold above this line:

```
<http security="none" pattern="/jss -sso/**" />
<http security="none" pattern="/logout.jsp" />
<http security="none" pattern="/secure/nosso.jsp" />
<http entry-point-ref="authenticationProcessingFilterEntryPoint">
```

Enabling manual (no SSO) login

SRC combines the login page with the main application, providing no manual (no SSO) login functionality. If you wish to enable manual login, allowing a user to see a login page when they click logout, follow these steps:

1. Rename the logout.jsp to logout.jsp.old.
2. Using your favourite text editor, create a new logout.jsp with the following content:

```
<% response.sendRedirect(request.getContextPath()+"/secure/nosso.jsp"); %>
```

3. Copy the src/secure/main.jsp to src/secure/nosso.jsp.
4. Rename the index.jsp to index.jsp.old.
5. Using your favourite text editor, create a new index.jsp with the following content:

```
<% response.sendRedirect(request.getContextPath()+"/secure/main.jsp"); %>
```

Accessing the application

When installation is complete, restart the Java web server and go to the SSO Plugin configuration interface to configure SSO: http://host/src/jss-sso/index.jsp

The default password for the configuration interface is jss.

After configuring SSO, use the Test SSO page to verify the username matches the usernames in the application and navigate the application homepage (http://host/src).

Configuring HP Asset Manager

Start by copying the SSO Plugin files to the target web application.

Patching application files

The application web.xml file must be patched by following the instructions in the section patching the web.xml file. The relevant patch file is web.xml.patch.am.

decorators.xml

Locate the decorators.xml file, found in the AM program directory, ie tomcat/webapps/Asset Manager/WEB-INF/cwc.

Add the following text in bold and save the file:

```
<excludes>
  <pattern>/jss-sso/**</pattern>
```

application-context.xml

Locate the application-context.xml file, found in the AM program directory, ie tomcat/webapps/Asset Manager/WEB-INF/classes/application-context.xml.

Open the file in a text editor, find the section below and add the text in bold:

```
   <property
name="convertUrlToLowerCaseBeforeComparison"><value>false</value></property
>
   <property name="publicResources">
     <list>
       <value>/jss-sso/**</value>
```

Accessing the application

When installation is complete, restart the Java web server and go to the SSO Plugin configuration interface to configure SSO: http://host/Asset_Manager/jss-sso/index.jsp

The default password for the configuration interface is jss.

After configuring SSO, use the Test SSO page to verify the username matches the usernames in the application and navigate the application homepage (http://host/Asset_Manager/index.jsp).

Configuring Kinetic Request

Kinetic Request 5.0.x is not compatible with SSO Plugin 3.6+. Please use SSO Plugin 3.5.x with KR 5.0.x, and SSO Plugin 3.6+ with KR 5.1+. You can find more information on the Kinetic Request SSO Plugin adapter on their website.

Start by copying the SSO Plugin files to the target web application. When configuring the SSO Plugin Authentication Service, enter the value kinetic.request.manual.login into the "Manual login key" field.

Patching application files

The application web.xml file must be patched by following the instructions in the section patching the web.xml file. The relevant patch file is web.xml.patch.kr.

Installing the SSO adapter from Kinetic

This information is correct at the time of writing, but the Kinetic package provides more detailed instructions. Kinetic will supply two files for this integration: JSSAuthenticator.jar and JSSAuthenticator.properties.

1. Copy the JSSAuthenticator.jar file into the Kinetic WEB-INF/lib directory.
2. Copy the JSSAuthenticator.properties file into the Kinetic WEB-INF/classes directory.

The package provided by Kinetic contains detailed instructions on the various property values and configuration information, which is also summarised in this document.

Configuring Kinetic Request

Navigate to: http://host/kinetic/AdminConsole and login as an administrator. Make the following changes:

1. Under properties, enter com.kd.kineticSurvey.authentication.JSSAuthenticator into the SSO Adapter class field.
2. Under properties, enter the full pathname to the JSSAuthenticator.properties file in the SSO Adapter properties field.

Restart Tomcat.

Accessing the application

When installation is complete, restart the Java web server and go to the SSO Plugin configuration interface to configure SSO: http://host/kinetic/jss-sso/index.jsp

The default password for the configuration interface is jss.

After configuring SSO, use the Test SSO page to verify the username matches the usernames in the application.

To make a survey accessible with SSO, select "Allow anonymous" in the Advanced tab, and under the Audit tab, select requires authentication and external in the drop down selector.

Finally, navigate to the survey and it should be accessible with SSO.

Configuring Kinetic Calendar

Start by copying the SSO Plugin files to the target web application.

Patching application files

The application web.xml file must be patched by following the instructions in the section patching the web.xml file. The relevant patch file is web.xml.patch.kc.

KinCal.xml

Locate the KinCal.xml file, which is located in the tomcat/webapps/KinCal/WEB-INF/classes directory.

Add the following text in bold:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
  <entry key="AuthenticatePublicPage">true</entry>
```

Restart Tomcat.

Accessing the application

When installation is complete, restart the Java web server and go to the SSO Plugin configuration interface to configure SSO: http://host/KinCal/jss-sso/index.jsp

The default password for the configuration interface is jss.

After configuring SSO, use the Test SSO page to verify the username matches the usernames in the application.

Finally, navigate to Kinetic Calendar and it should be accessible with SSO.

Configuring Jasper Reports 4.1/4.5/5

This integration has no dependency on BMC or HP ITSM. A separate integration document demonstrates how to configure Jasper Reports when integration with these products is required, allowing users and groups to automatically be synchronised between ITSM and Jasper Reports.

Start by copying the SSO Plugin files to the target web application.

Additionally, copy the jss-sso.jar file from the jasperserver directory into the target web application WEB-INF/lib directory, overwriting the file that was copied when copying the SSO Plugin files.

Patching application files

The application web.xml file must be patched by following the instructions in the section patching the web.xml file. The relevant patch file is web.xml.patch.jasper.

applicationContext-security.xml

Locate the file jasperreports/apache-tomcat/webapps/jasperserver/WEB-INF/applicationContext-security.xml.

Open the file in a text editor, find the section below and add the text in bold:

```
<beans …>
  <bean id="jss.j2eefilter"
class="com.javasystemsolutions.integrations.spring.security.SSOPluginPreAut
hFilter">
    <property name="authenticationManager"><ref
local="authenticationManager"/></property>
  </bean>
  <bean id="jss.preAuthenticatedAuthenticationProvider"
class="org.springframework.security.providers.preauth.PreAuthenticatedAuthe
nticationProvider">
    <property name="preAuthenticatedUserDetailsService">
      <bean
class="com.javasystemsolutions.integrations.jasper.SSOPluginUserDetailServi
ce">
        <property name="sessionFactory" ref="sessionFactory" />
        <property name="userAuthorityService"
ref="jss.txproxy.userAuthorityService" />
        <property name="tenantService" ref="jss.txproxy.tenantService" />
        <property name="tenantMap"><map>
          <entry key="localdomain.local" value="Test" />
        </map></property>
      </bean>
    </property>
  </bean>
  <bean id="jss.txproxy.tenantService"
class="org.springframework.transaction.interceptor.TransactionProxyFactoryB
ean">
    <property name="transactionManager" ref="transactionManager"/>
    <property name="target" ref="${bean.hibernateTenantService}"/>
    <property name="transactionAttributes"><props>
      <prop key="get*">PROPAGATION_REQUIRED</prop>
      <prop key="*">PROPAGATION_SUPPORTS</prop>
    </props></property>
  </bean>
  <bean id="jss.txproxy.userAuthorityService"
class="org.springframework.transaction.interceptor.TransactionProxyFactoryB
ean">
    <property name="transactionManager" ref="transactionManager"/>
    <property name="target" ref="${bean.internalUserAuthorityService}"/>
    <property name="transactionAttributes"><props>
      <prop key="get*">PROPAGATION_REQUIRED</prop>
      <prop key="put*">PROPAGATION_REQUIRES_NEW</prop>
      <prop key="update*">PROPAGATION_REQUIRES_NEW</prop>
      <prop key="*">PROPAGATION_SUPPORTS</prop>
    </props></property>
  </bean>
```

Find the following, and add the text in bold:

```
  <bean id="authenticationManager"
class="org.springframework.security.providers.ProviderManager">
    <property name="providers">
      <list>
        <ref bean="jss.preAuthenticatedAuthenticationProvider"/>
```

applicationContext-security-web.xml

Locate the file jasperreports/apache-tomcat/webapps/jasperserver/WEB-INF/applicationContext-security-web.xml.

Open the file in a text editor, find the section below and add the text in bold:

```
  <bean id="filterChainProxy"
class="org.springframework.security.util.FilterChainProxy">
    <property name="filterInvocationDefinitionSource">
      <value>
        CONVERT_URL_TO_LOWERCASE_BEFORE_COMPARISON
        PATTERN_TYPE_APACHE_ANT
        ...
/**=httpSessionContextIntegrationFilter,jss.j2eefilter,multipartRequestWrap
perFilter,...
      </value>
    </property>
  </bean>
```

In the same area, remove references to JIAuthenticationSynchronizer and a comma. It needs to be removed from each of the lines under <value>..</value>, ie. remove the text in bold:

```

/**=httpSessionContextIntegrationFilter,...JIAuthenticationSynchronizer,...
```

decorators.xml

Locate the file jasperreports/apache-tomcat/webapps/jasperserver/WEB-INF/decorators.xml.

Open the file in a text editor, find the section below and add the text in bold:

```
<excludes>
  <pattern>/jss-sso/*</pattern>
```

Mapping Windows domains to tenants

For organisations using the Jasper Reports enterprise edition, Windows domains can be mapped to tenants, mapping newly created users from a domain to a specific tenant. The tenants must exist and the following configuration (added above) is used to provide this mapping - each entry line maps a Windows DNS domain (the key) to the tenant (the value).

```
<property name="tenantMap"><map>
  <entry key="uk.corporate.com" value="UK" />
  <entry key="us.corporate.com" value="US" />
</map></property>
```

Accessing the application

When installation is complete, restart the Java web server and go to the SSO Plugin configuration interface to configure SSO: http://host/jasperserver/jss-sso/index.jsp

The default password for the configuration interface is jss.

After configuring SSO, use the Test SSO page to verify the username matches the usernames in the application.

Finally, navigate to Jasper Server with the following URL and SSO should take place: http://host:9080/jasperserver/

You can go directly to the login page by visiting this URL: http://host:9080/jasperserver/login.html

Configuring SAP Business Objects

Start by copying the SSO Plugin installation files to the BO application directory.

1. Stop the Tomcat instance running the Business Objects applications.
2. Locate the BOXI web application directory:
    a. In BOXI 3.x, locate the InfoViewApp web application directory, typically found in C:\Program Files\Business Objects\Tomcat55\webapps\InfoViewApp directory.
    b. In BOXI 4.x, locate the BOE web application directory, typically found in C:\Program Files\Business Objects\Tomcat55\webapps\BOE directory.
3. Copy the SSO Plugin files to the web application directory.
4. Start the Tomcat instance running the Business Objects applications.

Enabling SSO within BO

The following steps outline the integration process.

1. Locate the CMC Name. This is found in CMC -> Settings -> Cluster -> CMC Name and is often host.CentralManagementServer.
2. Login to the CMC and go to the Authentication, Enterprise, and tick 'Trusted Authentication is enabled'. Enter a unique value into the 'Shared secret' field, such as **somerandomvalue** for the purposes of this installation document. Press update.
3. Stop the Tomcat instance running the Business Objects applications.
4. The application web.xml file must be patched by following the instructions in the section patching the web.xml file. The relevant patch file is web.xml.patch.boxi.

    Further to the patching instructions, if manually patching, the patch is placed in the following locations depending on the version of BOXI:

    In BOXI 3.x, locate the last </context-param> element and paste after:

```
<context-param>
   <param-name>path.rightFrame</param-name>
   <param-value>1</param-value>
</context-param>
```

In BOXI 4.x, locate the ProxyApplicationLifeCycleListener and paste after:

```
<listener>
   <listener-
class>com.businessobjects.servletbridge.listener.ProxyApplicationLifeCycleL
istener</listener-class>
</listener>
```

5. A configuration change is required to further enable SSO in BOXI, but it differs per BOXI version.
    a. In BOXI 3.x, locate the following context-param section of the web.xml file:

```
<context-param>
  <param-name>trusted.auth.user.retrieval</param-name>
  <param-value></param-value>
</context-param>
```

and set the param-value to USER_PRINCIPAL, ie.

```
  <param-value>USER_PRINCIPAL</param-value>
```

Also locate:

```
<context-param>
  <param-name>sso.enabled</param-name>
  <param-value></param-value>
</context-param>
```

and set the param-value to true.

b. In BOXI 4.0, using your favourite text editor, create a file called custom.properties in the BOXI web application WEB-INF/config/custom directory and add the following content to the file:

```
sso.enabled=true
trusted.auth.user.retrieval=USER_PRINCIPAL
```

c. In BOXI 4.1, using your favourite text editor, create a file called global.properties in the BOXI web application WEB-INF/config/custom directory and add the following content to the file:

```
sso.types.and.order=trustedUserPrincipal
trusted.auth.user.retrieval=USER_PRINCIPAL
```

6. Create a file in the installation directory (C:\Program Files\Business Objects\BusinessObjects Enterprise 12.0\win32_x86) called TrustedPrincipal.conf with the following content that includes the random value from step 2:

```
SharedSecret=somerandomvalue
```

For other operating systems, the file is placed in the relevant directory as outlined below. Please note, the path will change depending on the version of BOXI installed and these paths are for guidance:

Windows 64bit: INSTALLDIR\SAP BusinessObjectsEnterprise XI 4.0\win64_x64\

AIX:	INSTALLDIR/sap_bobj/enterprise_xi40/aix_rs6000/
Solaris:	INSTALLDIR/sap_bobj/enterprise_xi40/solaris_sparc/
Linux:	INSTALLDIR/sap_bobj/enterprise_xi40/linux_x86

7. Start the Tomcat instance running the Business Objects applications.

Accessing the application

When installation is complete, go to the SSO Plugin configuration interface to configure SSO: http://host/InfoViewApp/jss-sso/index.jsp. The default password for the configuration interface is jss.

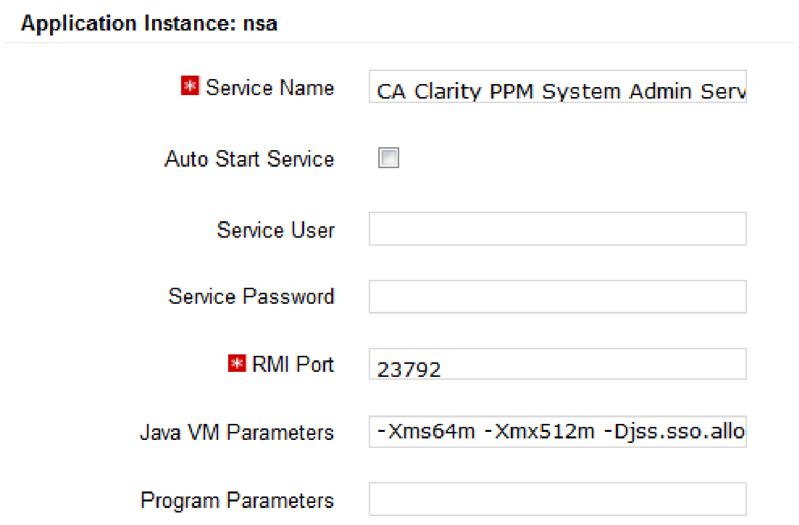
After using the SSO Plugin Test SSO function to ensure you have SSO access, and ensuring an account exists in BOXI with the same username, navigate to http://host/InfoViewApp/logon/logon.do on BOXI 3.x, and http://host/BOE/BI on BOXI 4.x.

Configuring CA Clarity PPM

Start by copying the SSO Plugin files to the target web application, ie the niku/webroot directory. However, delete the log4j-1.2.14.jar from niku/webroot/WEB-INF/lib after copying the files. Niku has its own version of log4j.jar in the niku/lib directory.

The following steps outline the integration process.

1. The application web.xml file must be patched by following the instructions in the section patching the web.xml file. The relevant patch file is web.xml.patch.ppm.
2. In the Clarity NSA application configuration, there are application parameter settings. Add the parameter -Djss.sso.allow.sso.http.parameter=true to the end as per the following screenshot.



Application Instance: nsa

Service Name	CA Clarity PPM System Admin Serv
Auto Start Service	☐
Service User	
Service Password	
RMI Port	23792
Java VM Parameters	-Xms64m -Xmx512m -Djss.sso.allo
Program Parameters	

3. Restart the Clarity server.
4. Navigate to the Clarity administration screen and locate the single-sign-on section.
   a. Locate the Token Name field and enter jss.sso.username
   b. Locate the Token Type selector and select Header.
   c. Locate and check the Use Single Sign-On checkbox.
5. Restart the Clarity server.

Enabling SSO Plugin logging

The Clarity application disables the SSO Plugin logging and provides its own configuration file. To enable SSO Plugin logging, locate the logging.xml in the config directory and add the following text in bold:

```
</category>
<category additivity="false" name="com.javasystemsolutions">
  <priority value="all"/>
  <appender-ref ref="STDOUT"/>
</category>
<root>
```

The log output will then appear in the X-ca.log and X-system.log files. You should only do this if reporting an issue and need to send the log file.

Accessing the application

When installation is complete, go to the SSO Plugin configuration interface to configure SSO: http://host/niku/jss-sso/index.jsp. The default password for the configuration interface is jss.

After using the SSO Plugin Test SSO function to ensure you have SSO access, and ensuring an account exists in Clarity with the same username, navigate to the default Clarity URL (http://host:8080/niku) and you should be logged in with SSO.

To access the administration interface, navigate to the default administration link and add ?sso=false, ie http://host:8090/niku/app?sso=false. This relies on the -Djss.sso.allow.sso.http.parameter=true JVM parameter being set up as discussed in the installation steps.

Configuring BMC MyIT 2.2+

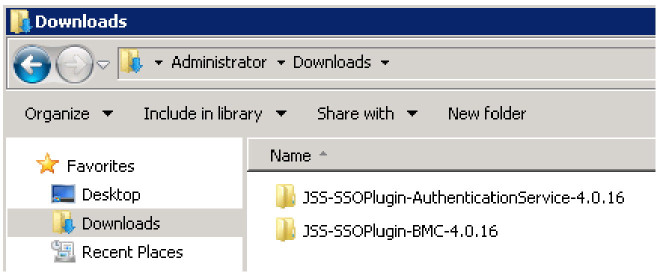
This installation method is used when you wish to deploy SSO directly into BMC MyIT, and not re-use an existing SSO implementation with BMC Mid Tier. For many clients, re-using an existing SSO integration on BMC Mid Tier is the fastest option to deployment, in which case please consult the Installation for BMC MyIT and SmartIT document.

If in any doubt, discuss your implementation options with JSS support.

Prerequisites

Two zip files are required for this installation: the SSO Plugin for BMC AR System and SSO Plugin Authentication Service. These can be found at the following URL: http://www.javasystemsolutions.com/jss/downloads

Example below with SSO Plugin v 4.0.16



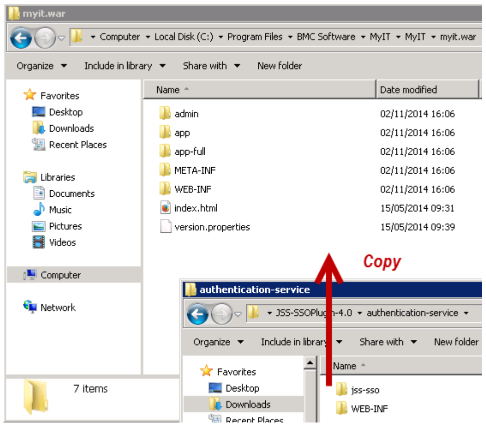
JSS-SSOPlugin-AuthenticationService-4.0.16
JSS-SSOPlugin-BMC-4.0.16

Enabling SSO within MyIT

Start by copying the SSO Plugin installation files to the BMC MyIT application directory.

1. Stop the Tomcat instance running MyIT.
2. Locate the ux directory, typically at C:\Program Files\BMC Software\Smart_IT_MyIT\Smart_IT_MyIT\ux.
3. Copy the SSO Plugin files to the ux directory.

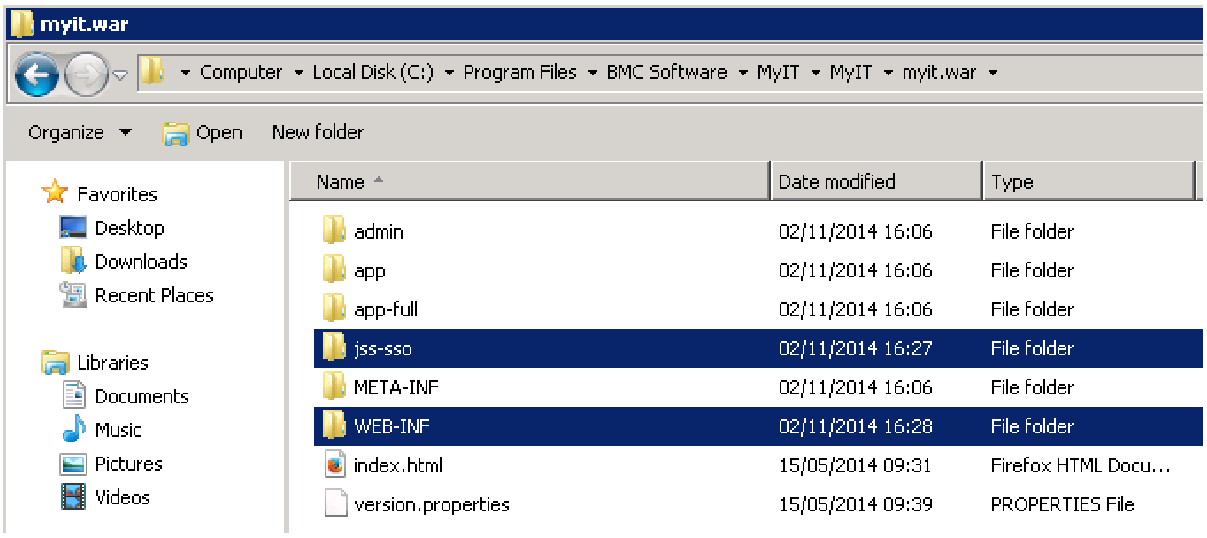
Example of copying the files



Name	Date modified
admin	02/11/2014 16:06
app	02/11/2014 16:06
app-full	02/11/2014 16:06
META-INF	02/11/2014 16:06
WEB-INF	02/11/2014 16:06
index.html	15/05/2014 09:31
version.properties	15/05/2014 09:39

jss-sso
WEB-INF

Example of the correct directory structure after copying files



myit.war

Computer ▸ Local Disk (C:) ▸ Program Files ▸ BMC Software ▸ MyIT ▸ MyIT ▸ myit.war

Name	Date modified	Type
admin	02/11/2014 16:06	File folder
app	02/11/2014 16:06	File folder
app-full	02/11/2014 16:06	File folder
jss-sso	02/11/2014 16:27	File folder
META-INF	02/11/2014 16:06	File folder
WEB-INF	02/11/2014 16:28	File folder
index.html	15/05/2014 09:31	Firefox HTML Docu...
version.properties	15/05/2014 09:39	PROPERTIES File

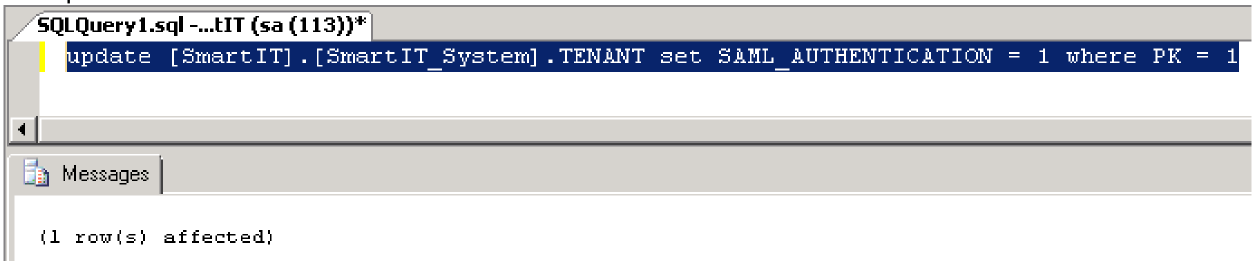
4. From the SSO Plugin for BMC AR System, locate the jss-sso-myit.jar from the myit/WEB-INF/lib directory and copy it into the MyIT installation WEB-INF/lib directory, ie C:\Program Files\BMC Software\MyIT\MyIT\myit.war\WEB-INF\lib.
5. Copy the MYIT web.xml patch file JSS-SSOPlugin-AuthenticationService-4.0.X\JSS-SSOPlugin-4.0\patches\web.xml.patch.myit into the web application WEB-INF directory and rename it web.xml.patch. Automatic patching will occur when SSO Plugin is configured through the web based user interface.
6. Locate the top level index.html file in the ux directory.
7. Rename it to index.jsp.
8. Open it in your favourite text editor and add the following to the top of the file:

```
<% response.setHeader("Cache-Control", "no-cache"); %>
```

9. A setting in the MyIT database switches on SSO. Locate the TENANT table and set SAML_AUTHENTICATION=1 where PK=1. Example SQL:

```
update [SmartIT]. [SmartIT_System].TENANT set SAML_AUTHENTICATION = 1 where
PK = 1
```

10. Example screenshot:



SQLQuery1.sql -...tIT (sa (113))*

```
update [SmartIT].[SmartIT_System].TENANT set SAML_AUTHENTICATION = 1 where PK = 1
```

```
(1 row(s) affected)
```

11. Start the instance of Tomcat running MyIT. The display name is **Smart IT/MyIT Application**

Configuring Jive Software 4.5+

Start by copying the SSO Plugin files to the target web application, keeping in mind that the Jive application root will be a path similar to /usr/local/jive/applications/template/application.

Unfortunately, Jive blocks the SSO Plugin configuration interface so the authentication-service.war file must be deployed to a Tomcat instance in order to configure the relevant SSO integration. The jss-ssoplugin.properties file can then be copied from the authentication service web application to the patched Jive application, typically placed in /usr/local/jive/applications/template/application/WEB-INF/classes.

The following steps outline the integration process - it is assumed [JIVE_HOME] points to the jive installation, typically /usr/local/jive.

1. The application web.xml file must be patched by following the instructions in the section patching the web.xml file. The relevant patch file is web.xml.patch.generic.
2. Locate the jss-sso-jive.jar file in the jive-software directory and copy it to the [JIVE_HOME] / applications/template/application/WEB-INF/lib directory.
3. Locate the Jive web.xml file, usually located in [JIVE_HOME] /applications/template/application/WEB-INF/web.xml. Make a backup of the file.
4. Open the web.xml file in your favourite text editor, locate and remove the following:

```
<listener>
  <listener-class>com.jivesoftware.community.aaa.JiveHttpSessionAttributeListener</listener-class>
</listener>
```

5. Create a directory [JIVE_HOME]/jss-sso.
6. Locate the jive-spring-jss.xml file in the jive-software directory and copy it to the [JIVE_HOME] /jss-sso directory.
7. Edit the [JIVE_HOME] /applications/sbs/bin/setenv file and locate the line:

```
# Place holder for custom application options
```

Place the following line after the line above:

```
CUSTOM_OPTS="-Djive.extensionPath=[JIVE_HOME]/jss-sso"
```

8. Restart the Jive server.

Configuring username integration

By default, the SSO username as configured through the Authentication Service configuration interface and reported through the Test SSO page, is used to query the Jive database for a matching user account.

For organisations with more complicated Jive usernames, such as user@dns.domain, a username format can be created using the user aliasing variables defined in the Configuring SSO Plugin document. The username format is configured in the jive-spring-jss.xml file:

```
<bean id="ssoPluginAuthenticationFilter"
class="com.javasystemsolutions.sso.integrations.jive.JiveSpringSecurityFilter" init-method="setup">
  ...
  <property name="userFormat" value="$SSO_USERNAME$@$SSO_DNS_DOMAIN$" />
```

Accessing the application

Jive does not provide an easy way to separate SSO and non-SSO, ie a single point of entry to access Jive using SSO. Therefore, the entire application is protected with SSO Plugin. If no matching Jive user is found in the Jive database, the user is presented with a login page.