Pentaho BI Server integration with Jasig CAS

Quick Installation Guide

Stratebi Business Solutions. (2016)

www.stratebi.com

Index

1 Introduction

This quick installation guide aims to deploy a ***P**entaho **B**usiness **I**ntelligence **S**erver (**BISERVER**)*, version **5.4.0.1 Community Edition**, in a local environment configured to use ***C**entral **A**uthentication **S**ervice (**CAS**)* protocol.

CAS provides access to a user to multiple systems with a single log-in process *(**S**ingle **si**gn-**o**n, **SSO***). It requires that all established communications are conducted through ***S**ecure **S**ockets **L**ayer (**SSL**)*, by which an **X.509** certificate will be also also needed for this deployment.

2 License

3 Requirements

3.1 Software

Software	Version	File size	Download Link
Pentaho Business Intelligence Server (BISERVER)	5.4.0.1-130	814,8 MB	biserver-ce-5.4.0.1-130.zip https://sourceforge.net/projects/pentaho/files/Business%20Intelligence%20Server/5.4/biserver-ce-5.4.0.1-130.zip/download
Apache Tomcat	6.0.45	7,0 MB	apache-tomcat-6.0.45.zip http://ftp.cixug.es/apache/tomcat/tomcat-6/v6.0.45/bin/apache-tomcat-6.0.45.zip
Jasig Central Authentication Service (JCAS)	3.5.3	32,3 MB	cas-server-webapp-3.5.3.war http://central.maven.org/maven2/org/jasig/cas/cas-server-webapp/3.5.3/cas-server-webapp-3.5.3.war
Jasig CAS Client (JCASC) For Java Core	3.1.12	85 KB	cas-client-core-3.1.12.jar http://central.maven.org/maven2/org/jasig/cas/client/cas-client-core/3.1.12/cas-client-core-3.1.12.jar
Spring Security CAS Support (SSCASS)	2.0.8	16 KB	spring-security-cas-client-2.0.8.RELEASE.jar http://central.maven.org/maven2/org/springframework/security/spring-security-cas-client/2.0.8.RELEASE/spring-security-cas-client-2.0.8.RELEASE.jar
OpenJDK	1.7	-	Depends on the operating system used.

3.2 Environment

Hardware - 64 bit	Operating System - 64 bit
Processor • Apple Macintosh Pro Quad-Core or Macintosh Mini Quad-Core. • Intel EM64T or AMD64 Dual-Core. **RAM** • 8 GB with 4 GB dedicated to Pentaho servers **Disk Space** • 20 GB free after installation	• Apple Macintosh OS X Server 10.9 & 10.10. • CentOS Linux 5 & 6. • Microsoft Windows 2008 Server R2, 2012 Server R2. • Red Hat Enterprise Linux 5 & 6. • Solaris 10. • Ubuntu Server 12.04 LTS & 14.04 LTS.

4 Environment Characteristics

This guide assumes the environment used has the following characteristics:

- The system account that will be use for this deployment **does not have administrator rights** and its name is **stratebi**.
- The host name is **pentaho**.
- The directory structure that will be use is as follows:
  1 **BASE**. Root directory. It may have any name.
    1.1 **APP**. Directory where application servers will be deployed.
    1.1 **ARCHIVE**. Backup directory. It might be needed.
    1.1 **TMP**. Temporal/trash directory.
    1.2 **SOURCE**. Directory where all the source files needed are located.
- Any command described here will use relative paths taking the **BASE** directory as the starting point.
- All references to files, will be described with relative paths using **BASE** directory as the starting point.
- All required ports used by the applications servers must be available and accessible:
  1 BISERVER: 8005, 8080, 8443, 8009.
  2 CAS Server: 80056 8088, 8444, 8010.

5 Installation Steps

In general, the sections to go through are the following:

1. Create a self-signed certificate to encrypt communications between the BISERVER and CAS Server. This certificate is needed for both servers.
2. Install Apache Tomcat with JCAS. A new **Connector** element will be created, which will be responsible for handling encrypted requests using the previously created self-signed certificate. Also, Tomcat's default ports need to be changed so it does not collide with BISERVER ports
3. Install BISERVER.
4. Install all components required to support CAS protocol on BISERVER.

Note that sections 1, 2 and 3 are used for a CAS Server deployment in a local environment, so if you already have a CAS Server installed, you might skip this part. Be aware that some of the configurations files present in this guide, will be pointing to a local CAS Server.

5.1 Create a X.509 certificate

The CAS implementation used in this guide, works only if it can be establish a secure connection with the client making the request. Therefore, an **X.509** certificate is required to enable such connections in Apache Tomcat. That being established, we will create a self-signed certificate that will be stored in a ***J**ava **K**ey**S**tore (**JKS**)* file type named **cas-certificate.jks**, an encrypted container used for certificate store. This container will be used by both, a CAS Server and a BISERVER..

5.1.1 Requirements

- Java Runtime Environment or Java Development Kit to create the self-signed certificate.

5.1.2 Creation Steps

1. Create the certificate and its container using the **keytool** provided by Java.

```
[stratebi@pentaho base]$                                                    Terminal
[stratebi@pentaho base]$ keytool -genkeypair -alias cas -keyalg RSA -dname "CN=localhost,OU=Quality &
Assurance,O=Stratebi,L=Madrid,S=MA,C=ES" -keypass password -keystore cas-certificate.jks -storepass
password
[invitado@CentOS casguide]$
```

2. Copy the file **cas-certificate.jks** in the **base > app** directory, where servers will be deployed.

5.2 Install CAS Server

J**asig **C**entral **A**uthentication **S**ervice (**JCAS**)*, is an implementation of the CAS protocol developed by Apereo's community. It is distributed as a web application in a ***WAR file *(**W**eb **A**pplication **A**rchive)*, so we will use Apache Tomcat as web server for its installation.

5.2.1 Requirements

- Java Runtime Environment or Java Development Kit to run Apache Tomcat server.
- Location of the container with the certificate that will be use to set up the secure connection to the CAS server where the JCAS reside.

5.2.2 Installation Steps

1. Download Apache Tomcat in the **source** folder.
2. Unzipm the **source > apache-tomcat-{version}.zip** file in the **app** folder. This creates a new directory named **apache-tomcat-{version}**.
3. Open Apache Tomcat configuration file, **app > apache-tomcat-{version} > conf > server.xml**, to change the default connection ports. This is necessary so that CAS server ports does not match with the biserver ones as this is deployed within its own Tomcat.

   (a) Modify the **port** attribute of the **Server** element, from 8005 to **8006** (line 22 approx.).

   ```
   22. <Server port="8006" shutdown="SHUTDOWN">
   23.   <Listener className="org.apache.catalina.startup.VersionLoggerListener" />
   ```

   (b) Modify the **port** and **redirectPort** attribute of the **Connector** element, from 8080 to **8088** and from 8443 to **8444** respectively (between lines 71 -. 73 approx).

   ```
   71. <Connector port="8088" protocol="HTTP/1.1"
   72.            connectionTimeout="20000"
   73.            redirectPort="8444" />
   ```

   (c) Configure the secure connection, by removing the tags **<!-- -->** surrounding the **Connector** element (between lines 83 -. 87 approx). Then, modify the **port** attribute of the **Connector** element, from 8443 to **8444** and add the following attributes: **keystoreFile="../cas-certificate.jks" keystorePass="password" keyAlias="cas"**. Note that the path used in the **keystoreFile** attribute is relative to the directory where Apache Tomcat has been deployed.

   ```
   83.
   84. <Connector port="8444" protocol="HTTP/1.1" SSLEnabled="true"
   85.            keystoreFile="../cas-certificate.jks" keystorePass="password" keyAlias="cas"
   86.            maxThreads="150" scheme="https" secure="true"
   87.            clientAuth="false" sslProtocol="TLS" />
   ```

   (d) Modify the **port** and **redirectPort** attribute of the **Connector** element from 8009 to **8010** and from 8443 to **8444** respectively to modify the port connection to the server through the AJP protocol (line 90 approx.).

   ```
   89. <!-- Define an AJP 1.3 Connector on port 8009 -->
   90. <Connector port="8010" protocol="AJP/1.3" redirectPort="8444" />
   ```

4. Save your changes and close the document.
5. Download JCAS, **cas-server-webapp-{versión}.war**, in **app > apache-tomcat-{version} > webapps** of Apache Tomcat. In the same moment you execute the Tomcat server, this will start to unzip the **cas-server-webapp-{versión}.war** file in the same directory where it was placed under the name of **cas-server-webapp-{versión}**, so not further steps are needed.

5.3 Install BISERVER

5.3.1 Requirements

- Java Runtime Environment or Java Development Kit to run BISERVER.

- CAS server up and running with certificate.
- Location of a trusted container with the CAS Server certificate in it.

5.3.2 Pasos de instalación

1. Download the BISERVER in the source folder.
2. Unzip the **source > biserver-ce-5.4.0.1-130.zip** file into **app** directory. This will create a new directory named **biserver-ce**.
3. Rename the newly created directory to **biserver-ce-5.4.0.1**. This helps you to remind which version of BISERVER we have installed.
4. Optionally, create a link between directories to refer to the latest version of BISERVER in use.

```
[stratebi@pentaho base]$ unzip ./source/biserver-ce-5.4.0.1-130.zip -d ./app/
[stratebi@pentaho base]$ mv ./app/biserver-ce/ ./app/biserver-ce-5.4.0.1
[stratebi@pentaho base]$ ln -s ./biserver-ce-5.4.0.1 app/biserver
[stratebi@pentaho base]$
```
Terminal

5. Duplicate the **app > biserver-ce-5.4.0.1 > start-pentaho.sh** file and rename the copy to **start-pentaho-ssl-trust.sh**, the latter will define a path to the trusted container with the CAS Server certificate in it that will be used when connecting to the CAS server. To do this, modify the **app > biserver-ce-5.4.0.1 > start-pentaho-ssl-trust.sh** file, and insert two new environment variables for the java virtual machine: **javax.net.ssl.trustStore**, which stores the file path where the trusted container is a **javax.net.ssl.trustStorePassword** that stores the password to access the stored certificate within the previously defined container (line 24 approx.).

```
21. ...
22. if [ "$errCode" = 0 ]; then
23.   cd "$DIR/tomcat/bin"
24.   CATALINA_OPTS="-Xms1024m -Xmx2048m -XX:MaxPermSize=256m
          -Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000
          -Djavax.net.ssl.trustStorePassword=password
          -Djavax.net.ssl.trustStore=/home/stratebi/base/app/cas-certificate.jks"
25.   export CATALINA_OPTS
26.   JAVA_HOME=$_PENTAHO_JAVA_HOME
27.   sh startup.sh
28. fi
```
Text

5.4 Install CAS Support for BISERVER

5.4.1 Requirements

- BISERVER previously installed.
- CAS Server web address: https://localhost:8444/cas-server-webapp-3.5.3/.

5.4.2 Installation Steps

1. Download **Jasig CAS Client** and **Spring Security CAS Support** in the source folder.
2. Copy the files **source > spring-security-cas-client-2.0.8.RELEASE.jar** and **source > cas-client-core-3.1.12.jar** into **app > biserver-ce-5.4.0.1 > tomcat > webapps > pentaho > WEB-INF > lib** folder with other Java libraries.

```
[stratebi@pentaho base]$
[stratebi@pentaho base]$ cp ./source/spring-security-cas-client-2.0.8.RELEASE.jar ./source/cas-client-core-3.1.12.jar app/biserver-ce-5.4.0.1/tomcat/webapps/pentaho/WEB-INF/lib
[stratebi@pentaho base]$
```
Terminal

3. Create the file **app > biserver-ce-5.4.0.1 > pentaho-solutions > system> applicationContext-spring-security-cas.xml** and include in it the content shown below. Pay particular attention to the highlighted lines:

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:pen="http://www.pentaho.com/schema/pentaho-system"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
```
Text

```
      http://www.pentaho.com/schema/pentaho-system http://www.pentaho.com/schema/pentaho-system.xsd" default-lazy-
init="true">

  <bean id="serviceProperties" class="org.springframework.security.ui.cas.ServiceProperties">
    <property name="service" value="http://localhost:8080/pentaho/j_spring_cas_security_check" />
    <property name="sendRenew" value="false" />
  </bean>

  <bean id="casProcessingFilter" class="org.springframework.security.ui.cas.CasProcessingFilter">
    <property name="authenticationManager" ref="authenticationManager" />
    <property name="authenticationFailureUrl" value="https://localhost:8444/cas-server-webapp-
3.5.3/authorizationFailure.jsp" />
    <property name="defaultTargetUrl" value="/" />
    <property name="filterProcessesUrl" value="/j_spring_cas_security_check" />
  </bean>

  <bean id="casProcessingFilterEntryPoint" class="org.springframework.security.ui.cas.CasProcessingFilterEntryPoint">
    <property name="loginUrl" value="https://localhost:8444/cas-server-webapp-3.5.3/login"/>
    <property name="serviceProperties" ref="serviceProperties" />
  </bean>

  <bean id="casAuthenticationProvider" class="org.springframework.security.providers.cas.CasAuthenticationProvider">
    <property name="userDetailsService">
      <pen:bean class="org.springframework.security.userdetails.UserDetailsService"/>
    </property>
    <property name="serviceProperties" ref="serviceProperties"/>
    <property name="ticketValidator">
      <bean class="org.jasig.cas.client.validation.Cas20ServiceTicketValidator">
        <constructor-arg index="0" value="https://localhost:8444/cas-server-webapp-3.5.3" />
      </bean>
    </property>
    <property name="key" value="an_id_for_this_auth_provider_only"/>
    <pen:publish as-type="org.springframework.security.providers.AuthenticationProvider">
      <pen:attributes>
        <pen:attr key="providerName" value="cas"/>
      </pen:attributes>
    </pen:publish>
  </bean>

   <bean id="filterChainProxy" class="org.springframework.security.util.FilterChainProxy">
    <property name="filterInvocationDefinitionSource">
      <value>
        <![CDATA[CONVERT_URL_TO_LOWERCASE_BEFORE_COMPARISON
        PATTERN_TYPE_APACHE_ANT

/webservices/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSess
ionContextIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFil
terForWS,filterInvocationInterceptorForWS

/api/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSessionConte
xtIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilterForWS
,filterInvocationInterceptorForWS

/plugin/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSessionCo
ntextIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilterFo
rWS,filterInvocationInterceptorForWS

/**=securityContextHolderAwareRequestFilter,httpSessionPentahoSessionContextIntegrationFilter,httpSessionContextIntegra
tionFilter,httpSessionReuseDetectionFilter,logoutFilter,casProcessingFilter,authenticationProcessingFilter,basicProcess
ingFilter,requestParameterProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilter,filterInvocationInterce
ptor]]>
      </value>
    </property>
  </bean>

  <bean id="authenticationManager" class="org.springframework.security.providers.ProviderManager">
    <property name="providers">
      <list>
        <ref bean="casAuthenticationProvider" />
        <ref bean="anonymousAuthenticationProvider" />
      </list>
    </property>
  </bean>

  <bean id="exceptionTranslationFilter" class="org.springframework.security.ui.ExceptionTranslationFilter">
    <property name="authenticationEntryPoint">
      <ref local="casProcessingFilterEntryPoint" />
    </property>
    <property name="accessDeniedHandler">
      <bean class="org.springframework.security.ui.AccessDeniedHandlerImpl" />
    </property>
  </bean>

  <bean id="exceptionTranslationFilterForWS" class="org.springframework.security.ui.ExceptionTranslationFilter">
    <property name="authenticationEntryPoint">
      <ref local="casProcessingFilterEntryPoint" />
```

```
    </property>
    <property name="accessDeniedHandler">
      <bean class="org.springframework.security.ui.AccessDeniedHandlerImpl" />
    </property>
  </bean>

  <bean id="logoutFilter" class="org.springframework.security.ui.logout.LogoutFilter">
    <constructor-arg value="https://localhost:8444/cas-server-webapp-3.5.3/logout?
url=http://localhost:8080/pentaho/Home" />
    <!-- URL redirected to after logout -->
    <constructor-arg>
      <list>
        <bean class="org.pentaho.platform.web.http.security.PentahoLogoutHandler" />
        <bean
            class="org.springframework.security.ui.logout.SecurityContextLogoutHandler" />
      </list>
    </constructor-arg>
    <property name="filterProcessesUrl" value="/Logout" />
  </bean>
</beans>
```

4. Modify the **app > biserver-ce-5.4.0.1 > pentaho-solutions > system > pentaho-spring-beans.xml** file adding a new entry immediately after **<import resource =" applicationContext-spring-security-jdbc.xml "/>** with the following content (line 91 approx.): **<import resource =" applicationContext-spring-security-cas.xml "/>**.

```
89. ...                                                                    Text
90. <import resource="applicationContext-pentaho-security-jdbc.xml" />
91. <import resource="applicationContext-spring-security-jdbc.xml" />
92. <import resource="applicationContext-spring-security-cas.xml" />
93.
```

5.5 Test

Once installation is completed, we must raise both servers and verify that access to biserver is satisfactory. To do this we must:

1. Start CAS server.
   (a) Go to **app > apache-tomcat-{version} > bin** directory and start the server using the **startup.sh** script.

```
[stratebi@pentaho base]$                                                   Terminal
[stratebi@pentaho base]$ cd ./app/apache-tomcat-6.0.45/bin
[stratebi@pentaho bin]$ ./startup.sh
Using CATALINA_BASE:   /home/stratebi/base/app/apache-tomcat-6.0.45
Using CATALINA_HOME:   /home/stratebi/base/app/apache-tomcat-6.0.45
Using CATALINA_TMPDIR: /home/stratebi/base/app/apache-tomcat-6.0.45/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/stratebi/base/app/apache-tomcat-6.0.45/bin/bootstrap.jar
[stratebi@pentaho bin]$
```

2. Start BISERVER.
   (a) Go to **app > biserver-ce-5.4.0.1** directory and start the server using the **start-pentaho-ssl-trust.sh** script. Then press the **ENTER** key to continue the boot process.

```
[stratebi@pentaho bin]$ cd -                                               Terminal
[stratebi@pentaho base]$ cd ./app/biserver-ce-5.4.0.1
[stratebi@pentaho biserver-ce-5.4.0.1]$ ./start-pentaho-ssl-trust.sh
WARNING: Using java from path
DEBUG: _PENTAHO_JAVA_HOME=
DEBUG: _PENTAHO_JAVA=java
-------------------------------------------------------------------------------------------
The Pentaho BI Platform now contains a version checker that will notify you
when newer versions of the software are available. The version checker is enabled by default.
For information on what the version checker does, why it is beneficial, and how it works see:
http://wiki.pentaho.com/display/ServerDoc2x/Version+Checker
Press Enter to continue, or type cancel or Ctrl-C to prevent the server from starting.
You will only be prompted once with this question.
-------------------------------------------------------------------------------------------
[OK]:

Using CATALINA_BASE:   /home/stratebi/base/app/biserver/tomcat
Using CATALINA_HOME:   /home/stratebi/base/app/biserver/tomcat
```

```
Using CATALINA_TMPDIR: /home/stratebi/base/app/biserver/tomcat/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/stratebi/base/app/biserver/tomcat/bin/bootstrap.jar
[stratebi@pentaho biserver-ce-5.4.0.1]$
```

3. Go to http://localhost:8080/ using your web browser of preference. This will make a connection request to the BISERVER, which then redirects the request to the CAS Server to perform authentication on its behalf: https://localhost:8444/cas-server-webapp-3.5.3/login?service=http%3A%2F%2Flocalhost%3A8080%2Fpentaho%2Fj_spring_cas_security_check
4. Given that the certificate used has not been signed by any recognizable authority by the web browser, we must add a security exception to continue to the login page of the CAS server.
5. Enter **admin** as username and password. This is possible because the CAS Server was installed in its default configuration so this, grants authorization to any credentials where the **user is equal to its password**. The CAS Server validates the credentials, grants an authorization token and redirects the web browser back to the BISERVER.
6. The BISERVER receives the authentication token, validates it against the CAS Server and redirects the browser to its home page, http://localhost:8080/pentaho/Home and checking in this sense, the correct configuration of the BISERVER with the CAS Server.

6 Extras

6.1 Files modified during installation

6.1.1 server.xml. Servidor CAS

Text

```
<?xml version='1.0' encoding='utf-8'?>
<!--
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->
<!-- Note:  A "Server" is not itself a "Container", so you may not
     define subcomponents such as "Valves" at this level.
     Documentation at /docs/config/server.html
 -->
<Server port="8006" shutdown="SHUTDOWN">

  <!--APR library loader. Documentation at /docs/apr.html -->
  <Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />
  <!--Initialize Jasper prior to webapps are loaded. Documentation at /docs/jasper-howto.html -->
  <Listener className="org.apache.catalina.core.JasperListener" />
  <!-- Prevent memory leaks due to use of particular java/javax APIs-->
  <Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />
  <!-- JMX Support for the Tomcat server. Documentation at /docs/non-existent.html -->
  <Listener className="org.apache.catalina.mbeans.ServerLifecycleListener" />
  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />

  <!-- Global JNDI resources
       Documentation at /docs/jndi-resources-howto.html
  -->
  <GlobalNamingResources>
    <!-- Editable user database that can also be used by
         UserDatabaseRealm to authenticate users
    -->
    <Resource name="UserDatabase" auth="Container"
              type="org.apache.catalina.UserDatabase"
              description="User database that can be updated and saved"
              factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
              pathname="conf/tomcat-users.xml" />
  </GlobalNamingResources>

  <!-- A "Service" is a collection of one or more "Connectors" that share
       a single "Container" Note:  A "Service" is not itself a "Container",
       so you may not define subcomponents such as "Valves" at this level.
```

```
      Documentation at /docs/config/service.html
  -->
 <Service name="Catalina">

   <!--The connectors can use a shared executor, you can define one or more named thread pools-->
   <!--
   <Executor name="tomcatThreadPool" namePrefix="catalina-exec-"
       maxThreads="150" minSpareThreads="4"/>
   -->


   <!-- A "Connector" represents an endpoint by which requests are received
        and responses are returned. Documentation at :
        Java HTTP Connector: /docs/config/http.html (blocking & non-blocking)
        Java AJP  Connector: /docs/config/ajp.html
        APR (HTTP/AJP) Connector: /docs/apr.html
        Define a non-SSL HTTP/1.1 Connector on port 8080
   -->
   <Connector port="8088" protocol="HTTP/1.1"
              connectionTimeout="20000"
              redirectPort="8444" />
   <!-- A "Connector" using the shared thread pool-->
   <!--
   <Connector executor="tomcatThreadPool"
              port="8080" protocol="HTTP/1.1"
              connectionTimeout="20000"
              redirectPort="8443" />
   -->
   <!-- Define a SSL HTTP/1.1 Connector on port 8443
        This connector uses the JSSE configuration, when using APR, the
        connector should be using the OpenSSL style configuration
        described in the APR documentation -->

   <Connector port="8444" protocol="HTTP/1.1" SSLEnabled="true"
              keystoreFile="cas-server.jks" keystorePass="password" keyAlias="serverkey"
              maxThreads="150" scheme="https" secure="true"
              clientAuth="false" sslProtocol="TLS" />

   <!-- Define an AJP 1.3 Connector on port 8009 -->
   <Connector port="8010" protocol="AJP/1.3" redirectPort="8444" />


   <!-- An Engine represents the entry point (within Catalina) that processes
        every request.  The Engine implementation for Tomcat stand alone
        analyzes the HTTP headers included with the request, and passes them
        on to the appropriate Host (virtual host).
        Documentation at /docs/config/engine.html -->

   <!-- You should set jvmRoute to support load-balancing via AJP ie :
   <Engine name="Catalina" defaultHost="localhost" jvmRoute="jvm1">
   -->
   <Engine name="Catalina" defaultHost="localhost">

     <!--For clustering, please take a look at documentation at:
         /docs/cluster-howto.html  (simple how to)
         /docs/config/cluster.html (reference documentation) -->
     <!--
     <Cluster className="org.apache.catalina.ha.tcp.SimpleTcpCluster"/>
     -->

     <!-- The request dumper valve dumps useful debugging information about
          the request and response data received and sent by Tomcat.
          Documentation at: /docs/config/valve.html -->
     <!--
     <Valve className="org.apache.catalina.valves.RequestDumperValve"/>
     -->

     <!-- This Realm uses the UserDatabase configured in the global JNDI
          resources under the key "UserDatabase".  Any edits
          that are performed against this UserDatabase are immediately
          available for use by the Realm.  -->
     <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
            resourceName="UserDatabase"/>

     <!-- Define the default virtual host
          Note: XML Schema validation will not work with Xerces 2.2.
      -->
     <Host name="localhost"  appBase="webapps"
           unpackWARs="true" autoDeploy="true"
           xmlValidation="false" xmlNamespaceAware="false">

       <!-- SingleSignOn valve, share authentication between web applications
            Documentation at: /docs/config/valve.html -->
       <!--
       <Valve className="org.apache.catalina.authenticator.SingleSignOn" />
       -->
```

```
       <!-- Access log processes all example.
            Documentation at: /docs/config/valve.html -->
       <!--
       <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
              prefix="localhost_access_log." suffix=".txt" pattern="common" resolveHosts="false"/>
       -->

     </Host>
   </Engine>
  </Service>
</Server>
```

6.1.2 pentaho-spring-beans.xml. BISERVER

Text

```
<?xml version='1.0' encoding='utf-8'?>
<!--+
  | This should be the only file specified in web.xml's contextConfigLocation. It should only contain imports.
  +-->

<beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:pen="http://www.pentaho.com/schema/pentaho-system"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
http://www.pentaho.com/schema/pentaho-system http://www.pentaho.com/schema/pentaho-system.xsd" default-lazy-
init="true">

<bean class="org.pentaho.platform.engine.core.system.objfac.spring.ApplicationContextPentahoSystemRegisterer"
scope="singleton"/>

  <bean id="SystemConfig" class="org.pentaho.platform.config.SystemConfig">
    <constructor-arg>
      <pen:list class="org.pentaho.platform.api.engine.IConfiguration"/>
    </constructor-arg>
    <pen:publish as-type="INTERFACES"/>
  </bean>

  <bean class="org.pentaho.platform.config.SolutionPropertiesFileConfiguration">
    <constructor-arg value="security"/>
    <constructor-arg value="security.properties"/>
    <pen:publish as-type="INTERFACES"/>
  </bean>

  <bean class="org.pentaho.platform.config.PentahoPropertyPlaceholderConfigurer" >
    <constructor-arg>
      <pen:bean class="org.pentaho.platform.api.engine.ISystemConfig"/>
    </constructor-arg>
  </bean>

  <bean class="org.pentaho.platform.config.SolutionPropertiesFileConfiguration">
    <constructor-arg value="system"/>
    <constructor-arg value="system.properties"/>
    <pen:publish as-type="INTERFACES"/>
  </bean>

  <bean class="org.pentaho.platform.config.SolutionPropertiesFileConfiguration">
    <constructor-arg value="sqlmetadataqueryexec"/>
    <constructor-arg value="sqlmetadataqueryexec.properties"/>
    <pen:publish as-type="INTERFACES"/>
  </bean>

  <import resource="pentahoSystemConfig.xml" />
  <import resource="adminPlugins.xml" />
  <import resource="systemListeners.xml" />
  <import resource="repository.spring.xml" />
  <import resource="applicationContext-spring-security.xml" />
  <import resource="applicationContext-spring-security-superuser.xml" />
  <import resource="applicationContext-pentaho-security-superuser.xml" />

  <import resource="applicationContext-common-authorization.xml" />
  <import resource="applicationContext-spring-security-memory.xml" />

  <import resource="applicationContext-pentaho-security-memory.xml" />
  <import resource="applicationContext-spring-security-ldap.xml" />
  <import resource="applicationContext-pentaho-security-ldap.xml" />

  <import resource="applicationContext-pentaho-security-jackrabbit.xml" />
  <import resource="applicationContext-spring-security-jackrabbit.xml" />
  <import resource="applicationContext-pentaho-security-jdbc.xml" />
  <import resource="applicationContext-spring-security-jdbc.xml" />
  <import resource="applicationContext-spring-security-cas.xml" />


  <import resource="pentahoObjects.spring.xml" />
  <import resource="GettingStartedDB-spring.xml" /> <!-- Remove this line to unhook the Getting Started DB -->
```

```
  <import resource="importExport.xml" />
  <import resource="defaultUser.spring.xml"/>
  <import resource="sessionStartupActions.xml" />
  <import resource="olap4j.spring.xml"/>
</beans>
```

6.1.3 applicationContext-spring-security-cas.xml. BISERVER

```
<?xml version='1.0' encoding='utf-8'?>

<beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:pen="http://www.pentaho.com/schema/pentaho-system"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
       http://www.pentaho.com/schema/pentaho-system http://www.pentaho.com/schema/pentaho-system.xsd" default-lazy-
init="true">

  <bean id="serviceProperties" class="org.springframework.security.ui.cas.ServiceProperties">
    <property name="service" value="http://localhost:8080/pentaho/j_spring_cas_security_check" />
    <property name="sendRenew" value="false" />
  </bean>

  <bean id="casProcessingFilter" class="org.springframework.security.ui.cas.CasProcessingFilter">
    <property name="authenticationManager" ref="authenticationManager" />
    <property name="authenticationFailureUrl" value="https://localhost:8444/cas-server-webapp-
3.5.3/authorizationFailure.jsp" />
    <property name="defaultTargetUrl" value="/" />
    <property name="filterProcessesUrl" value="/j_spring_cas_security_check" />
  </bean>

  <bean id="casProcessingFilterEntryPoint" class="org.springframework.security.ui.cas.CasProcessingFilterEntryPoint">
    <property name="loginUrl" value="https://localhost:8444/cas-server-webapp-3.5.3/login"/>
    <property name="serviceProperties" ref="serviceProperties" />
  </bean>

  <bean id="casAuthenticationProvider" class="org.springframework.security.providers.cas.CasAuthenticationProvider">
    <property name="userDetailsService">
      <pen:bean class="org.springframework.security.userdetails.UserDetailsService"/>
    </property>
    <property name="serviceProperties" ref="serviceProperties"/>
    <property name="ticketValidator">
      <bean class="org.jasig.cas.client.validation.Cas20ServiceTicketValidator">
        <constructor-arg index="0" value="https://localhost:8444/cas-server-webapp-3.5.3" />
      </bean>
    </property>
    <property name="key" value="an_id_for_this_auth_provider_only"/>
    <pen:publish as-type="org.springframework.security.providers.AuthenticationProvider">
      <pen:attributes>
        <pen:attr key="providerName" value="cas"/>
      </pen:attributes>
    </pen:publish>
  </bean>

   <bean id="filterChainProxy" class="org.springframework.security.util.FilterChainProxy">
    <property name="filterInvocationDefinitionSource">
      <value>
        <![CDATA[CONVERT_URL_TO_LOWERCASE_BEFORE_COMPARISON
        PATTERN_TYPE_APACHE_ANT

/webservices/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSess
ionContextIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFil
terForWS,filterInvocationInterceptorForWS

/api/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSessionConte
xtIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilterForWS
,filterInvocationInterceptorForWS

/plugin/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSessionCo
ntextIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilterFo
rWS,filterInvocationInterceptorForWS

/**=securityContextHolderAwareRequestFilter,httpSessionPentahoSessionContextIntegrationFilter,httpSessionContextIntegra
tionFilter,httpSessionReuseDetectionFilter,logoutFilter,casProcessingFilter,authenticationProcessingFilter,basicProcess
ingFilter,requestParameterProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilter,filterInvocationIntercep
tor]]>
      </value>
    </property>
  </bean>

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

```
    </property>
  </bean>

  <bean id="exceptionTranslationFilter" class="org.springframework.security.ui.ExceptionTranslationFilter">
    <property name="authenticationEntryPoint">
      <ref local="casProcessingFilterEntryPoint" />
    </property>
    <property name="accessDeniedHandler">
      <bean class="org.springframework.security.ui.AccessDeniedHandlerImpl" />
    </property>
  </bean>

  <bean id="exceptionTranslationFilterForWS" class="org.springframework.security.ui.ExceptionTranslationFilter">
    <property name="authenticationEntryPoint">
      <ref local="casProcessingFilterEntryPoint" />
    </property>
    <property name="accessDeniedHandler">
      <bean class="org.springframework.security.ui.AccessDeniedHandlerImpl" />
    </property>
  </bean>

  <bean id="logoutFilter" class="org.springframework.security.ui.logout.LogoutFilter">
    <constructor-arg value="https://localhost:8444/cas-server-webapp-3.5.3/logout?
url=http://localhost:8080/pentaho/Home" />
    <!-- URL redirected to after logout -->
    <constructor-arg>
      <list>
        <bean class="org.pentaho.platform.web.http.security.PentahoLogoutHandler" />
        <bean
            class="org.springframework.security.ui.logout.SecurityContextLogoutHandler" />
      </list>
    </constructor-arg>
    <property name="filterProcessesUrl" value="/Logout" />
  </bean>
</beans>
```

6.1.4 start-pentaho-ssl-trust.sh. BISERVER

Text

```
#!/bin/sh
### ====================================================================== ###
##                                                                          ##
##  Pentaho Start Script                                                    ##
##                                                                          ##
### ====================================================================== ###

DIR_REL=`dirname $0`
cd $DIR_REL
DIR=`pwd`
#cd -

. "$DIR/set-pentaho-env.sh"

setPentahoEnv "$DIR/jre"
errCode=0
if [ -f "$DIR/promptuser.sh" ]; then
  sh "$DIR/promptuser.sh"
  errCode="$?"
  rm "$DIR/promptuser.sh"
fi
if [ "$errCode" = 0 ]; then
  cd "$DIR/tomcat/bin"
  CATALINA_OPTS="-Xms1024m -Xmx2048m -XX:MaxPermSize=256m -Dsun.rmi.dgc.client.gcInterval=3600000
-Dsun.rmi.dgc.server.gcInterval=3600000 -Djavax.net.ssl.trustStorePassword=password
-Djavax.net.ssl.trustStore=/home/invitado/casguide/base/app/biserver-ce-5.4.0.1/cas-server-truststores.jks"
  export CATALINA_OPTS
  JAVA_HOME=$_PENTAHO_JAVA_HOME
  sh startup.sh
fi
```

6.2 Logs

6.2.1 catalina.out. Log of the first run of the BISERVER

Log File

```
Mar 29, 2016 11:10:00 AM org.apache.catalina.core.AprLifecycleListener init
INFO: The APR based Apache Tomcat Native library which allows optimal performance in production
environments was not found on the java.library.path:
/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib
Mar 29, 2016 11:10:02 AM org.apache.coyote.http11.Http11Protocol init
INFO: Initializing Coyote HTTP/1.1 on http-8080
Mar 29, 2016 11:10:02 AM org.apache.catalina.startup.Catalina load
INFO: Initialization processed in 3738 ms
Mar 29, 2016 11:10:03 AM org.apache.catalina.core.StandardService start
```

```
INFO: Starting service Catalina
Mar 29, 2016 11:10:03 AM org.apache.catalina.core.StandardEngine start
INFO: Starting Servlet Engine: Apache Tomcat/6.0.43
Mar 29, 2016 11:10:03 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory ROOT
Mar 29, 2016 11:10:04 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory sw-style
Mar 29, 2016 11:10:05 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory pentaho-style
Mar 29, 2016 11:10:05 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory pentaho
[Server@1037449b]: [Thread[main,5,main]]: checkRunning(false) entered
[Server@1037449b]: [Thread[main,5,main]]: checkRunning(false) exited
[Server@1037449b]: Initiating startup sequence...
[Server@1037449b]: Server socket opened successfully in 7 ms.
[Server@1037449b]: Database [index=0, id=0, db=file:../../data/hsqldb/sampledata, alias=sampledata] opened
sucessfully in 3938 ms.
[Server@1037449b]: Database [index=1, id=1, db=file:../../data/hsqldb/hibernate, alias=hibernate] opened
sucessfully in 92 ms.
[Server@1037449b]: Database [index=2, id=2, db=file:../../data/hsqldb/quartz, alias=quartz] opened
sucessfully in 131 ms.
[Server@1037449b]: Startup sequence completed in 4173 ms.
[Server@1037449b]: 2016-03-29 11:10:18.075 HSQLDB server 1.8.0 is online
[Server@1037449b]: To close normally, connect and execute SHUTDOWN SQL
[Server@1037449b]: From command line, use [Ctrl]+[C] to abort abruptly
Attempting to load ESAPI.properties via file I/O.
Attempting to load ESAPI.properties as resource file via file I/O.
Not found in 'org.owasp.esapi.resources' directory or file not readable: /home/stratebi/base/app/biserver-
ce-5.4.0.1/tomcat/bin/ESAPI.properties
Not found in SystemResource Directory/resourceDirectory: .esapi/ESAPI.properties
Not found in 'user.home' (/home/stratebi) directory: /home/stratebi/esapi/ESAPI.properties
Loading ESAPI.properties via file I/O failed. Exception was: java.io.FileNotFoundException
Attempting to load ESAPI.properties via the classpath.
SUCCESSFULLY LOADED ESAPI.properties via the CLASSPATH from '/ (root)' using current thread context class
loader!
SecurityConfiguration for Validator.ConfigurationFile not found in ESAPI.properties. Using default:
validation.properties
Attempting to load validation.properties via file I/O.
Attempting to load validation.properties as resource file via file I/O.
Not found in 'org.owasp.esapi.resources' directory or file not readable: /home/stratebi/base/app/biserver-
ce-5.4.0.1/tomcat/bin/validation.properties
Not found in SystemResource Directory/resourceDirectory: .esapi/validation.properties
Not found in 'user.home' (/home/stratebi) directory: /home/stratebi/esapi/validation.properties
Loading validation.properties via file I/O failed.
Attempting to load validation.properties via the classpath.
validation.properties could not be loaded by any means. fail. Exception was:
java.lang.IllegalArgumentException: Failed to load ESAPI.properties as a classloader resource.
Pentaho BI Platform server is ready. (Pentaho Open Source BA Server 5.4.0.1-130) Fully Qualified Server
Url = http://localhost:8080/pentaho/, Solution Path = /home/stratebi/base/app/biserver-ce-5.4.0.1/pentaho-
solutions
Mar 29, 2016 11:17:08 AM org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8080
Mar 29, 2016 11:17:09 AM org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:8009
Mar 29, 2016 11:17:09 AM org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=0/193  config=null
Mar 29, 2016 11:17:09 AM org.apache.catalina.startup.Catalina start
INFO: Server startup in 426472 ms
```

6.2.2 catalina.out. Log de la primera corrida del Servidor CAS

```
Log File
Apr 06, 2016 10:35:04 AM org.apache.catalina.core.AprLifecycleListener init
INFO: The APR based Apache Tomcat Native library which allows optimal performance in production
environments was not found on the java.library.path:
/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib
Apr 06, 2016 10:35:05 AM org.apache.coyote.http11.Http11Protocol init
INFO: Initializing Coyote HTTP/1.1 on http-8088
Apr 06, 2016 10:35:07 AM org.apache.coyote.http11.Http11Protocol init
INFO: Initializing Coyote HTTP/1.1 on http-8444
Apr 06, 2016 10:35:07 AM org.apache.catalina.startup.Catalina load
INFO: Initialization processed in 3745 ms
Apr 06, 2016 10:35:07 AM org.apache.catalina.core.StandardService start
INFO: Starting service Catalina
Apr 06, 2016 10:35:07 AM org.apache.catalina.core.StandardEngine start
INFO: Starting Servlet Engine: Apache Tomcat/6.0.45
Apr 06, 2016 10:35:07 AM org.apache.catalina.startup.HostConfig deployDescriptor
```

```
INFO: Deploying configuration descriptor manager.xml
Apr 06, 2016 10:35:08 AM org.apache.catalina.startup.HostConfig deployDescriptor
INFO: Deploying configuration descriptor host-manager.xml
Apr 06, 2016 10:35:09 AM org.apache.catalina.startup.HostConfig deployWAR
INFO: Deploying web application archive cas-server-webapp-3.5.3.war
2016-04-06 10:35:25,391 INFO [org.jasig.cas.services.DefaultServicesManagerImpl] - <Loaded 1 services.>
2016-04-06 10:35:26,157 WARN
[org.jasig.cas.authentication.handler.support.SimpleTestUsernamePasswordAuthenticationHandler] -
<org.jasig.cas.authentication.handler.support.SimpleTestUsernamePasswordAuthenticationHandler is only to
be used in a testing environment.  NEVER enable this in a production environment.>
2016-04-06 10:35:30,606 INFO [org.jasig.cas.util.AutowiringSchedulerFactoryBean] - <Starting Quartz
Scheduler now>
Apr 06, 2016 10:35:35 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory ROOT
Apr 06, 2016 10:35:35 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory docs
Apr 06, 2016 10:35:36 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory examples
Apr 06, 2016 10:35:37 AM org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8088
Apr 06, 2016 10:35:37 AM org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8444
Apr 06, 2016 10:35:38 AM org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:8010
Apr 06, 2016 10:35:38 AM org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=0/147  config=null
Apr 06, 2016 10:35:38 AM org.apache.catalina.startup.Catalina start
INFO: Server startup in 31050 ms
2016-04-06 10:35:44,103 INFO [org.jasig.cas.ticket.registry.support.DefaultTicketRegistryCleaner] -
<Beginning ticket cleanup.>
2016-04-06 10:35:44,118 INFO [org.jasig.cas.ticket.registry.support.DefaultTicketRegistryCleaner] - <0
tickets found to be removed.>
2016-04-06 10:35:44,119 INFO [org.jasig.cas.ticket.registry.support.DefaultTicketRegistryCleaner] -
<Finished ticket cleanup.>
2016-04-06 10:37:25,755 INFO [org.jasig.cas.services.DefaultServicesManagerImpl] - <Reloading registered
services.>
2016-04-06 10:37:25,756 INFO [org.jasig.cas.services.DefaultServicesManagerImpl] - <Loaded 1 services.>
```

7 About Stratebi

Stratebi Business Solutions is specialist in Business Intelligence, Analytics, Big Data, Liferay, Open Source Solutions, Java development and Digital Transformation Contact us at:

web: ww.stratebi.com

email: info@stratebi.com

phone: (+34) 91.788.34.10

These are some samples of our front end work


General
Efficiency
Economics
Performance
Savings Indicators
Consumption / Type of appliance
Energetic Status
Consumption / Hour type