HITACHI
Inspire the Next

Pentaho Server SAML Authentication with Hybrid Authorization

This page intentionally left blank.

Contents

This page intentionally left blank.

Overview

SAML is a specification that provides a means to exchange an authentication assertion of the principal (user) between an identity provider (IdP) and a service provider (SP). Once the plugin is built and installed, your Pentaho Server will become a SAML service provider, relying on the assertion from the IdP to provide authentication.

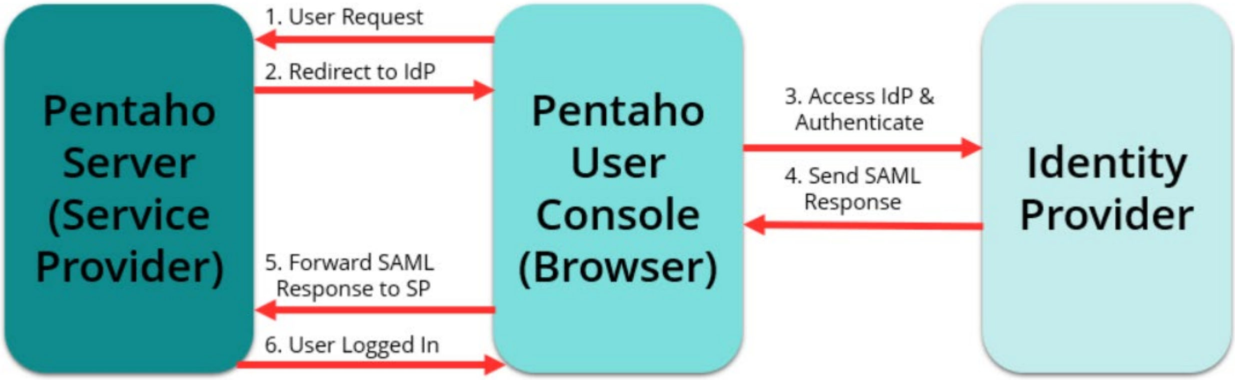
Pentaho Server (Service Provider)
1. User Request
2. Redirect to IdP
5. Forward SAML Response to SP
6. User Logged In
Pentaho User Console (Browser)
3. Access IdP & Authenticate
4. Send SAML Response
Identity Provider

Figure 1: SAML and Pentaho

The intention of this document is to speak about topics generally; however, these are the specific versions covered here:

Software	Version(s)
Pentaho (BA Server) Enterprise Edition	7.x, 8.x

The Components Reference in Pentaho Documentation has a complete list of supported software and hardware.

Before You Begin

Before beginning, use the following information to prepare for the procedures described in the main section of the document.

Terms You Should Know

Here are some terms you should be familiar with:

- **Identity Provider (IdP)**: issues authentication assertions in conjunction with a single sign-on (SSO) profile of SAML
- **Service Provider (SP)**: receives and accepts authentication assertions from the IdP
- **Keystore**: stores all keys necessary for SAML

Other Prerequisites

Before you get started, you will need to:

- Select an IdP
- Have Pentaho Enterprise Edition installed
- Decide on and set up an authorization method for user role assignment
- Build a Karaf archive that will be installed in the Pentaho Server:
  - Download the Engineering Samples Repository zip file (whichever is the right branch for your Pentaho version)
  - Compile the sample

SAML capabilities are currently provided as source code in an engineering sample and are therefore not distributed by Pentaho in a binary form.

Install and Configure the SAML Plugin

This section will guide you through the steps of installing and configuring the SAML plugin files for the Pentaho Server. More information on these topics is available here:

- Step 1: Unpacking and Installing the SAML Plugin
- Step 2: Creating Your Own SAML Certificate and Keystore
- Step 3: Preparing the Pentaho Service Provider File

Step 1: Unpacking and Installing the SAML Plugin

First, you will need to unpack the SAML files and place them in the correct directories. Once the plugin is installed, the `pentaho.saml.cfg` file will be created in the `$SERVER_HOME/pentaho-solutions/system/karaf/etc` directory:

1. Verify that the Pentaho Server has started by checking the `catalina.log` located in the `tomcat/logs` directory.
2. Make sure that you have these three files in your SAML file package:
   a. `pentaho-saml-sample.kar`
   b. `applicationContext-spring-security-saml.xml`
   c. `logout.jsp`
3. Place the `pentaho-saml-sample.kar` file into this directory: `pentaho-solutions/system/karaf/deploy`
4. Stop the Pentaho Server.
5. Place the `applicationContext-spring-security-saml.xml` file into the `pentaho-solutions/system` directory.
6. Copy the `logout.jsp` into the `$SERVER_HOME/tomcat/webapps/pentaho` directory.

Step 2: Creating Your Own SAML Certificate and Keystore

You may create your own SAML assertion signing and encryption certificate or obtain a signed certificate from a certificate authority. If you do obtain a certificate, make sure that it uses a hash algorithm supported by your IdP, such as SHA-1 or SHA-256.

Here is the process of creating your own SAML certificate and keystore. Make sure to change the directory path, along with the passwords for the `storepass` and the `keypass` shown in red in the example below. These passwords do not need to be the same.

1. Navigate to the `$SERVER_HOME` directory and create a folder named `saml`.
2. Open a terminal or command prompt and make `$SERVER_HOME/saml` your working directory.
3. Run the `keytool` command to generate a self-signed certificate.

```
$PENTAHO_JAVA_HOME/bin/keytool -genkey -alias saml -keystore
$SERVER_HOME/saml/saml.keystore.jks -storepass changeit -keyalg RSA –
keypass changeit
```

Step 3: Preparing the Pentaho Service Provider File

Creating the SP metadata is usually a prerequisite to obtaining identity provider metadata from the IdP service of choice.

If you do not already have an SP metadata file, download this sample into a Unix-formatted file called `pentaho-sp.xml` for editing.

This section describes how to modify a template SP metadata file to match your Pentaho Server installation:

1. Move or copy your SP metadata file to the `$SERVER_HOME/saml` folder.

Moving the SP metadata file to this location makes it easier to follow along in these directions. If you move the file elsewhere, modify these instructions accordingly.

2. Locate the tag entries for `<md:SingleLogoutService>` and `<md:AssertionConsumerService>` and replace the values of the `Location` attribute with the correct protocol, hostname, and port for your environment.
3. Export the contents of your `saml` signing certificate (and additional encryption certificate if you generated one) to a `base64` representation using the following `keytool` command:

```
$PENTAHO_JAVA_HOME/bin/keytool -exportcert -keystore
$SERVER_HOME/saml/saml.keystore.jks -storepass changeit -alias saml -rfc
```

   You should get certificate data that looks like this:

```
-----BEGIN CERTIFICATE----
CERTIFICATE
DATA
PAYLOAD
-----END CERTIFICATE-----
```

The use attribute of the parent <md:KeyDescriptor> tag defines if you are dealing with the signing or encryption certificate. The same certificate data can be used in both spots.

4. Copy the content of the certificate data payload, omitting the `BEGIN` and `END` lines, into the appropriate `<ds:X509Certificate>` entry tag.

Configure Your IdP for Use with Pentaho

This section walks you through the basic steps needed to configure your chosen IdP for SAML.

First, get the IdP metadata file by following these steps:

1. Provide your Pentaho SP XML file to your IdP administrator.
2. Have your IdP administrator register Pentaho as a service provider with the IdP.
3. Get your IdP metadata XML file from your IdP administrator.

Depending on your IdP and its configuration and requirements, your IdP administrator may also provide you with certificates used for communicating with the IdP.

If you *do not* need to import certificates into your SAML keystore, skip to the next section, Configure the Pentaho SAML File.

If you *do* need to import those certificates:

1. Get the certificates from your IdP administrator.
2. Open a CMD prompt.
3. Replace the `keystore` passwords and paths in this example command, then run it from a terminal:

```
$PENTAHO_JAVA_HOME/bin/keytool -import -alias saml -keystore
$SERVER_HOME/saml/saml.keystore.jks -storepass changeit -file
$SERVER_HOME/saml/saml.signing.cer
```

Configure the Pentaho SAML File

After you have the IdP metadata XML file, you will need to edit the `pentaho.saml.cfg` file to configure the IdP to work with SAML. At this point, you should have the following:

- A `saml` folder with an IdP metadata XML file
- A SP metadata XML file
- A `keystore` file

When editing the `pentaho.saml.cfg` file, note that absolute paths (no variables) must be listed in the file.

In the examples below, we use `$SERVER_HOME=/pentaho` to represent the absolute path.

The SP, IdP, and `keystore` files can be referenced using filesystem, URL, or classpath locations. Using the filesystem properties (`saml.idp.metadata.filesystem`, `saml.sp.metadata.filesystem`, and `saml.keystore.filesystem`) is the recommended method. Only one property of each file type should be enabled at any time, and the unused properties should be commented out.

You can find more information on the following topics here:

- Step 1: Setting the IdP Properties
- Step 2: Setting the SP Properties
- Step 3: Setting the Keystore Properties
- Step 4: IdP Specific Configuration
- Step 5: Enabling the SAML Plugin
- Step 6: Setting Up Hybrid Role Assignment

Step 1: Setting the IdP Properties

The first thing you will need to do is to set the `saml.idp.url` property which is used to select the proper `EntityDescriptor` from the referenced `saml.idp.xml` file:

1. Navigate to the `$SERVER_HOME/saml` directory and open the `idp.xml` file.
2. Locate the `<EntityDescriptor>` tag and copy the value of the `entityID` attribute.

```
<EntityDescriptor ID="..." entityID="http://the-idp-url" xmlns=
```

3. Set the value of the `saml.idp.url` property to the copied `entityID` value.
4. Verify that the entries for `saml.idp.metadata.url` and `saml.idp.metadata.classpath` are commented out.
5. Make sure that the entry for `saml.idp.metadata.filesystem` is uncommented.
6. Change the path for the `saml.idp.metadata.filesystem` to match the path of your IdP metadata XML file:

```
saml.idp.metadata.filesystem=/pentaho/saml/idp.xml
```

Step 2: Setting the SP Properties

This example will use the filesystem method, since earlier instructions directed you to save the `pentaho-sp.xml` file.

*The `saml.sp.metadata.entityId` property defaults to a value of `pentaho`. This value must match the `entityID` in the SP metadata XML file and the party trust configured in the IdP. Editing this value is **not** recommended.*

1. Verify that the entries for `saml.sp.metadata.url` and `saml.sp.metadata.classpath` are commented out.
2. Make sure the entry for `saml.sp.metadata.filesystem` is uncommented.
3. Change the path for the `saml.sp.metadata.filesystem` to match the path of your SP metadata XML file:

```
saml.sp.metadata.filesystem=/pentaho/saml/pentaho-sp.xml
```

Step 3: Setting the Keystore Properties

All the certificates needed for signing, encryption, and communication with SAML servers need to be in a single `keystore` file.

1. Verify that the entries for the `saml.keystore.url` and `saml.keystore.classpath` properties are commented out.
2. Make sure the entry for `saml.keystore.filesystem` is uncommented.
3. Change the value of `saml.keystore.filesystem` to:

```
saml.keystore.filesystem=/pentaho/saml/saml.keystore.jks
```

4. Locate the `saml.keystore.default.key` property and change it to match the alias of your saml signing certificate. The path set up before referenced a self-signed certificate aliased as `saml`, which would be configured as:

```
saml.keystore.default.key=saml
```

5. Locate and set the `keystore` password with the `saml.keystore.password` property. This should match the password used as the `-storepass` argument when you obtained a SAML signing certificate.

```
saml.keystore.password=changeit
```

6. If any of your certificate private key passwords include the colon : character, change the `saml.username.password.delimiter.char` property to a valid delimiter character not included in any of the key passwords.

7. Provide a comma-separated list of `alias<delimiter>password` to allow the Pentaho SAML plugin to read private keys in the `saml.keystore.private.username.passwords` property:

```
saml.keystore.private.username.passwords=saml:changeit,saml2:changeit
```

The `saml.keystore.private.username.passwords` property refers to `username`, which is equivalent to `key alias`.

Step 4: IdP Specific Configuration

Set the following properties in `pentaho.saml.cfg` to match the configuration of your IdP:

Table 1: `pentaho.saml.cfg` Properties

Property	Value
`use.global.logout.strategy`	`true`
`ensure.incoming.logout.request.signed`	`false`
`ensure.outgoing.logout.response.signed`	`true`
`ensure.outgoing.logout.request.signed`	`true`

Step 5: Enabling the SAML Plugin

To enable the SAML plugin:

1. Shut down the Pentaho Server, if it is running.
2. Locate the `$SERVER_HOME/pentaho-solutions/system` directory and open the `pentaho-spring-beans.xml` file with any text editor.
3. Find the line that references `<import resource="applicationContext-spring-security-jdbc.xml"./>` and add this line directly beneath it:

```
<import resource="applicationContext-spring-security-saml.xml" />
```

4. Save and close the `pentaho-spring-beans.xml` file.
5. In the same directory, locate the `security.properties` file and open it.
6. Change the `provider` value in the top line to `saml`:

```
provider=saml
```

At this point, you may start the Pentaho Server to test with native role assignment, which allows you to test and receive the `Authenticated` role.

Step 6: Setting Up Hybrid Role Assignment

The use of either JDBC or LDAP hybrid role assignment (authorization) is required for all functions of the Pentaho User Console (PUC) to work. The `authorization.provider` property allows you to choose the authorization method to delegate to, while SAML will be used for user authentication.

1. Set the value of `authorization.provider` in `pentaho.saml.cfg` to the selected role provider (generally `ldap` or `jdbc`).
2. Start the Pentaho Server to test.

Tips, Common Issues, and Debugging

Here are some configuration and troubleshooting tips for SAML:

Disabling the SAML Plugin

If you decide to disable the SAML plugin, reverse the changes that you made to enable it by:

1. Shutting down the Pentaho Server, if it is running.
2. Commenting out the line added that imports `applicationContext-spring-security-saml.xml` that was added to the `pentaho-spring-beans.xml` file.
3. Changing the provider value in the `security.properties` file back to a non-`saml` value, such as `jackrabbit`, `jdbc`, or `ldap`.

Clearing and Installing a New SAML Plugin KAR File

When changes are made, and a new KAR file is produced, it must be redeployed to the Pentaho Server. To perform a redeploy, you must clear the Karaf cache, remove the previously deployed KAR file, and restart the Pentaho Server.

This will create a new `pentaho-solutions/system/karaf/etc/pentaho.saml.cfg` file, so you should move your old file, let the plugin installation process create a new file, and then compare the contents of the files to see what you might need to change in the new file.

1. Shut down the Pentaho Server.
2. Delete the `data` subfolders of the server's `$SERVER_HOME/pentaho-solutions/system/karaf/caches/default/`.
3. Delete the SAML KAR file from the server's `$SERVER_HOME/pentaho-solutions/system/karaf/deploy/`.
4. Add the new SAML KAR file in the same directory.
5. Start the Pentaho Server and wait for Karaf to rebuild its cache and deploy/install the new KAR.
6. Shut down the Pentaho Server.
7. Start the Pentaho Server again.

Checking for Common Issues Made Editing pentaho.saml.cfg

- Ensure there is only one active setting specifying external configuration files (`saml.sp.metadata`, `saml.idp.metadata`, `saml.keystore`) By default, the classpath version is active, but the recommended configuration is filesystem. Ensure the other methods are disabled with the `#` at the beginning of their property line.
- Verify the value of the `saml.sp.metadata.entityId` property matches the `entityId` specified in the root tag of the SP metadata.

Installing JCE Unlimited Strength Security in Your JRE

This step is required so the JVM can use larger key sizes for your certificate. Without installing JCE Unlimited Strength, you may notice `InvalidKeyException` errors in the Pentaho Server Tomcat logs.

1. Download the appropriate JCE Unlimited Strength archive for the version of the Java Runtime you have hosting Tomcat for your Pentaho Server.
2. Follow the instructions to install, which will be packaged with the download in a `README` file.

Enabling Debug Logging

To enable debug logging:

1. Open `$SERVER_HOME/tomcat/webapps/Pentaho/WEB-INF/log4j.xml`.
2. Add the following `log4j` directives:

```
<category name="org.springframework.security">
 <priority value="INFO"/>
</category>

<category name="org.springframework.security.saml">
 <priority value="DEBUG"/>
</category>

<category name="org.pentaho.platform.spring.security.saml">
 <priority value="DEBUG" />
</category>

<category name="org.opensaml">
 <priority value="DEBUG" />
</category>
```

3. Restart the Pentaho Server.

Related Information

Here are some links to information that you may find helpful while using this best practices document:

- GitHub: Samples for Extending Pentaho
- GitHub: Sample for SAML
- Pentaho Components Reference
- Pentaho: LDAP Security
- Pentaho: Manual LDAP/JDBC Hybrid Configuration

Finalization Checklist

This checklist is designed to be added to any implemented project that uses this collection of best practices, to verify that all items have been considered and reviews have been performed.

Name of the Project:____

Date of the Review:____

Name of the Reviewer:____

Item	Response	Comments
Did you build and install the SAML plugin?	YES_______ NO_______	
Did you create your own SAML signing certificate and keystore?	YES_______ NO_______	
Did you prepare the Pentaho SP file?	YES_______ NO_______	
Did you have your IdP administrator register Pentaho as a service provider?	YES_______ NO_______	
Did your IdP administrator give you the IdP metadata XML file?	YES_______ NO_______	
Did you import any certificates required by your IdP?	YES_______ NO_______	
If necessary, did you install JCE Unlimited Strength in JRE?	YES_______ NO_______	
Did you set the SP properties?	YES_______ NO_______	
Did you set the IdP properties?	YES_______ NO_______	
Did you set the keystore properties?	YES_______ NO_______	
Did you set up any IdP specific configuration?	YES_______ NO_______	
Did you enable the SAML plugin?	YES_______ NO_______	
Did you change the `authorization.provider` property?	YES_______ NO_______	