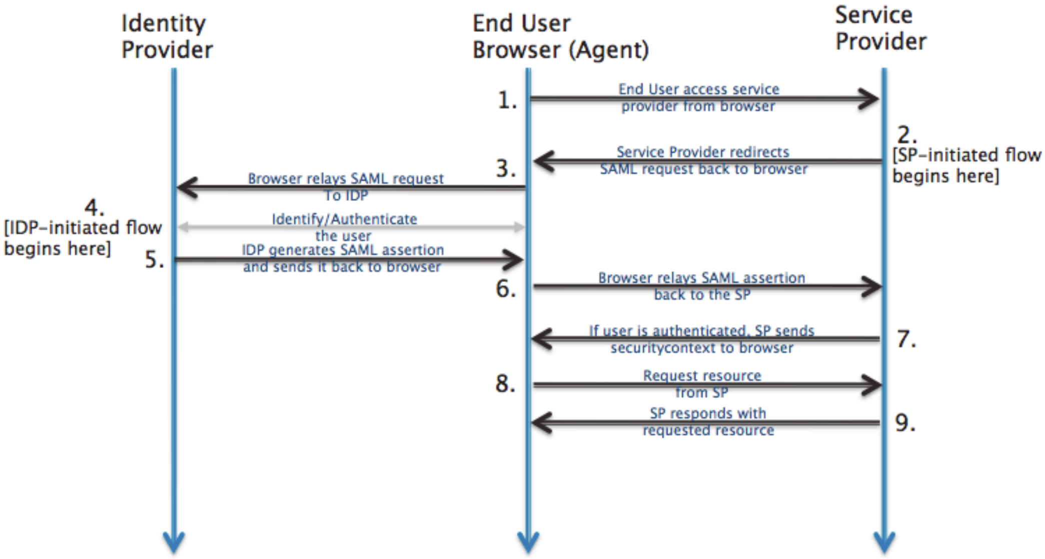
Registering in Salesforce-developer IdP

SAML: Overview

[1] SAML is mostly used as a web-based authentication mechanism as it relies on the browser being used as an agent that brokers the authentication flow. At high-level, the authentication flow of SAML looks like this:


Identity Provider
End User Browser (Agent)
Service Provider
1.
End User access service provider from browser
2.
[SP-initiated flow begins here]
3.
Service Provider redirects SAML request back to browser
Browser relays SAML request To IDP
4.
[IDP-initiated flow begins here]
Identify/Authenticate the user
5.
IDP generates SAML assertion and sends it back to browser
6.
Browser relays SAML assertion back to the SP
7.
If user is authenticated, SP sends securitycontext to browser
8.
Request resource from SP
9.
SP responds with requested resource

Identity Provider	developer.okta.com	The 3rd-party entity that takes care of the user's authentication; There are multiple such entities with SAML protocol support, such as OpenSSO, SSOCircle.com, OneLogin.com, Salesforce.com… For this example, we'll be using Salesforce.com
End-user browser agent	Pentaho User	User that accesses BA-server via browser
Service Provider	BA-server	Pentaho BA Server

[1] http://developer.okta.com/docs/guides/saml_guidance.html

Registering in Salesforce-developer Identification Provider (IdP)

Note: Each user that intends to use BA-server needs to register itself first in Salesforce-developer

Prerequisites

1. **<u>Have your chosen Service Provider (i.e. Ba-Server) metadata xml file at hand</u>**

   <u>Developer/QA only</u>: if none is created yet, you can leverage on the already existing SP metadata file for Pentaho BA-Server. For this:
   a. Next to this document, you should have a "`resources`" folder
   b. Download "pentaho-sp.xml" and rename it to something more identifiable with salesforce (e.g. "pentaho-salesforce-dev-sp.xml");

2. **<u>Have a certificate file holding the public signing key at hand</u>**

   <u>Developer/QA only</u>: if none is created yet, we can quickly create one based of the information stored if the service provider's metadata xml file:
   a. Create a new (empty) file with extension .cer (e.g. "pentaho-dev-certificate.cer")
   b. Open it using an editor of your choice (Notepad++, Sublime Text, ..)
   c. Add the following content **<u>as-is</u>** to the file:

```
-----BEGIN CERTIFICATE-----
<certificate-goes-here>
-----END CERTIFICATE-----
```

   d. Open your Service provider (i.e. Ba-Server) metadata xml file using an editor of your choice (Notepad++, Sublime Text, ..)
   e. Locate the `<ds:X509Certificate>` tag
      i. If two such tags exist, we want the one that is inside the `<md:KeyDescriptor use="signing">` tag and not the one inside the `<md:KeyDescriptor use="encryption">`
      ii. Copy its contents **<u>as-is (do not change/edit anything)</u>**
      iii. Paste it in the newly created certificate file, replacing the placeholder `<certificate-goes-here>` with this pasted content
      iv. **<u>Roughly put</u>**, your certificate should look something like this:

```
-----BEGIN CERTIFICATE-----
MIIDUjCCAjqgAwINBgCQYDVQQGEwJGSTEQMA4GA1UE
CBMHVXVzaW1hYTEAoTD1JNNSBTb2Z0d2FyZSBPeTEM
MAoGA1bG8wHhcNMTMwMTAxMTExWhcNMjIxMjMwMTEy
ODAxWjBrMQswMHVVRMA8G1UEBxMGVs7BlLYW/GuHE=
-----END CERTIFICATE-----
```

Registering yourself in developer.salesforce.com

1. Go to https://developer.salesforce.com/
2. Click “Sign Up >” on near the top right corner of the page
    a. Fill in all fields
        i. Be sure to provide a real email, as a confirmation email will be sent to you.
    b. Click on “Sign me up >”
3. Go to your e-mail and finish the registration
4. Once registration is done, login to https://salesforce.com/
5. After login, go to “Setup” on the top menu, right of your user name
6. On the left menu, above “Build”, expand “Create” and click on “Apps”
7. Below “Save” and “Cancel” buttons on the top, there should be the following text: “To publish an app, you need to have chosen a namespace prefix. Click here to choose a namespace prefix. “
    a. Click on “Click here to choose a namespace prefix.”
8. In “Change Developer Setting” page
    a. After “NameSpace Prefix”, write a domain prefix (e.g., pentahodevtest)
    b. Click on “Check Availability” button
    c. If the text: “This namespace is available” appears, then the prefix you choose is not owned and you can continue, otherwise choose a different prefix name
    d. “Package to be managed:” should be “--None--“
    e. Click on “Review My Selections”
9. You will go to a new page where you can review your “Namespace Prefix”
    a. Click “Save”
10. You will be re-directed to your home page
    a. Go to “Setup” on the top menu, right of your user name
11. On the left menu, above “Administer”, expand “Domain Management” and click on “My Domain”
12. You will go to the “My Domain” page
    a. Type a domain name after “https://” (e.g., pentahodevtest)
    b. Your idp url will be:
    ”https://{the domain you typed}-dev-ed.my.salesforce.com/”
    c. Click on “Check Availability” button
    d. If the text: “Available” appears, then the domain name you choose is not owned and you can continue, otherwise choose a different domain name
    e. Check “I agree to the Terms and Conditions”
    f. Click the “Register Domain” button
    g. After, you will have to wait for the domain to be accepted, it should not take long
        i. You will receive an e-mail from salesforce.com
13. Go to “Setup” on the top menu, right of your user name

14. On the left menu, above “Administer”, expand “Domain Management” and click on “My Domain”
15. After the text: “Your domain name is available for testing.”
    a. Click on the “Click here to login” button
    b. The page will reload and bellow the button you have just clicked you should be able to click on “Deploy To Users” button
    c. Please click on “Deploy to Users” button
        i. The page will ask you to confirm the operation because you can not revert it.
            1. Click “OK”

Registering a Service Provider in developer.salesforce.com

1. Go to “Setup” on the top menu, right of your user name
2. On the left menu, above “Build”, expand “Create” and click on “Apps”
3. You will go to the “Apps” page
    a. After “Connected Apps”
    b. Click on “New” button
4. In the “New Connected App” page, you will insert the required information about the SP
    a. Above “Basic Information”, please follow the table below to fill the fields, if a field is not mention leave it empty

Field Name	**Field value**
Connected App Name	pentaho
API Name	pentaho
Contact Email	{your e-mail}

    b. Above “Web App Settings”, please follow the table below to fill the fields, if a field is not mention leave it empty

Field Name	**Field value**
Start URL	http://{sp server ip:port or name}/pentaho (e.g., http://localhost:8080/)
Enable SAML	Checkbox: checked
Entity Id	pentaho
ACS URL	http://{sp server ip:port or name}/pentaho/saml/SSO (e.g., http://localhost:8080/pentaho/saml/SSO)
Subject Type	Username
Name ID Format	urn:osasis:names:tc:SAML:1.1:nameid-format-unspecified
Issuer	{Empty}
Verify Request Signatures	Checkbox: checked
Encrypt SAML Response	Checkbox: checked
Block Encryption Algorithm	AES-256

        i. For both “Encrypt SAML Response” and “Encrypt SAML Response”, below each, there is a button “Choose File”
            1. For both, click on the button “Choose File” and select your public key exported by the SP
                a. If you don’t have a public key, only for QA/Dev/Services, please follow the instructions in step 2 of section “Prerequisites”
    c. Click “Save”
5. You will go to your newly create App page
    a. Click on “Manage” button
6. In the “Manage” page of your App, you will need to allow users to access your application, to do this you can associate Profiles to your app or set a Permission Sets, in this tutorial, you will associate a Profile:
    a. After the text “Profiles”, click on the button “Manage Profiles”
        i. From the list of profiles, select at least “System Administrator” which is the profile of your user, you should enable other profiles, so common users can access the app
        ii. Click “Save”

Creating Pentaho Roles and assigning them to Users

In this section, you will create a salesforce role, please pay attention that the **user can only have one role**

1. Go to “Setup” on the top menu, right of your user name
2. On the left menu, above “Administer”, expand “Manage Users” and click on “Roles”
3. On the “Understanding Roles” page
    a. Click on “Set Up Roles” button
4. On the “Creating the Role Hierarchy” page
    a. Just below your company name (e.g., pentaho), which should be the root of the Hierarchy
    b. Click on “Add Role”
5. On the “New Role” page
    a. Define the role name that can be later used as a Pentaho Role
        i. The best way to do this is to add a prefix to the group name itself (for example, “Pentaho:”)
        ii. So, for example, if the role would be “Administrator”, the role name would be “Pentaho:Administrator”
        iii. **Important**: **you are free to choose the prefix you desire; please memorize the prefix you have chosen, as you will need to reference it afterwards in pentaho.saml.cfg, in the “saml.role.related.user.attribute.prefix” property**
    b. Write the role name for “Label” and “Role Name” fields
        i. If you use ‘:’ in the role name, it will be replaced by ‘_’ on the “Role Name”, there will be no problems with that
    c. Click on “Save”
6. You will be redirect to the role page, you can now add users

    a. Click on “Assign Users to Role” button

7. In this new page
    a. Above the text “Available Users Search:”, select “All Users”
    b. Click on “Find” button
    c. Click on the user name that you want to add to this role, in order to select it
    d. Above “Add”, click on “>”
    e. Repeat the two last steps to add more users
    f. When done, click the “Save” button

Sending Pentaho Roles alongside the Authentication Credentials

1. Go to “Setup” on the top menu, right of your user name
2. On the left menu, above “Build”, expand “Create” and click on “Apps”
3. You will go to the “Apps” page
    a. Below “Connected Apps” and on the left of your app’s name (e.g., pentaho)
    b. Click on “Manage”
4. You will go to the “Manage” page of your app
    a. On the bottom after “Custom Attributes” text
    b. Click on “New” button
5. In the “Create Custom Attribute” page, you will be able to create an attribute
    a. After “Attribute key”, type the name of the attribute that will carry the role (e.g., “Pentaho Role”)
    b. **Important**: **you are free to choose the name you desire; please memorize the name you have chosen, as you will need to reference it afterwards in pentaho.saml.cfg, in the “saml.role.related.user.attribute.name” property**
    c. Click on “Insert Field” button
        i. The “Insert Field” window will appear
        ii. On the left column click on “$UserRole >”
        iii. On the right column click on “Name”
        iv. After, click on the “Insert” button
    d. The window will close
    e. Click on “Save” button

Getting Salesforce metadata xml file

1. Go to “Setup” on the top menu, right of your user name
2. On the left menu, above “Build”, expand “Create” and click on “Apps”
3. You will go to the “Apps” page
    a. Below “Connected Apps” and on the left of your app’s name (e.g., pentaho)
    b. Click on “Manage”
4. You will go to the “Manage” page of your app
    a. Click on “Donwload Metadata” button
5. Click on the “Download Metadata” button
    a. **Save this xml metadata file in your local machine**.
    b. **This is Salesforce providing us a auto-generated “Salesforce IdP Metadata” xml.**

c. Rename it to something that will help you identify it (example: “salesforce-metadata-idp.xml”)

d. **Important**: **you will need to place the path to this file afterwards in pentaho.saml.cfg, in the “saml.idp.metadata.filesystem” property**

e. Open “salesforce -metadata-idp.xml” with a text editor of your choice

f. Locate the “entityID” attribute

   i. It should be something like “https://{the domain you choosed}-dev-ed.my.salesforce.com/”

   ii. **Important**: **copy-paste this value into pentaho.saml.cfg, in the “saml.idp.url” property**

Setting pentaho-solutions/system/karaf/etc/pentaho.saml.cfg properties

1. Edit pentaho-solutions/system/karaf/etc/pentaho.saml.cfg
2. Locate property **saml.idp.metadata.filesystem**
   a. Set the path to the Salesforce metadata xml file you downloaded in previous steps
3. Locate property **saml.idp.url**
   a. Open your Salesforce metadata xml file with a text editor of your choice
   b. Locate the “entityID” attribute
      i. It should be something like “https://{the domain you choosed}-dev-ed.my.salesforce.com/”
      ii. Copy-paste that value into the saml.idp.url property
4. Locate property **saml.role.related.user.attribute.name**
   a. Set the name of the attribute that carries the Roles we’ve created in previous steps (e.g., “Pentaho Role”)
5. Locate property **saml.role.related.user.attribute.prefix**
   a. Set the prefix (if one was defined) that each of the Pentaho Roles will hold (e.g., “Pentaho:”)

Recap

We have:

1. Registered onto developer.salesforce.com

2. Added an application called “pentaho”, uploaded the our SP public key, passed the endpoint for it (`http://localhost:8080/pentaho/saml/SSO`) and have configured it to work with SAML 2.0

3. Created Salesforce “role” that start with a “Pentaho:” prefix and assigned this to our user

4. Configured the SAML response so that the authenticated response (from Salesforce to Pentaho) also carries an attribute called “Pentaho Role” with the name of the salesforce role the user has

5. Got the idp and sp metadata xml files

6. Got the Salesforce url to use from the “entityID” value

Q&A

Q1 | Do I need a certificate to sign the authentication requests?

Yes.

For this sample, we are using a certificate provided by spring-security-saml, stored in a .jks (keystore file).

It's already bundled in the saml-authentication-provider sample (jar:/security/keystore.jks).

You can get the original here: https://github.com/spring-projects/spring-security-saml/blob/1.0.1.RELEASE/core/src/test/resources/org/springframework/security/saml/key/keystore.jks

This certificate is used by some IdP's, such as SSOCircle.com and okta.developer.com.

If you plan to connect to some other IdPs, then you must ensure you update the keystore file to include the certificate provided by that Identification Provider.