Configuring Sakai 11 for SAML Authentication with ADFS

DRAFT, 5 Jan 2017, Stephen Marquard, stephen.marquard@uct.ac.za

Introduction

Sakai 11 adds support for SAML authentication using Spring Security SAML (https://jira.sakaiproject.org/browse/SAK-30105).

This example shows how to configure Sakai to use Microsoft ADFS as an Identification Provider (IdP) for Sakai. This assumes that ADFS is only used to authenticate users in Sakai who already have accounts in Sakai, i.e. it does not deal with creating accounts dynamically.

This example is based on the installation at the University of Cape Town:

- The ADFS IdP is adfs.uct.ac.za
- The development Sakai 11 system is devslscle001.uct.ac.za
- The production Sakai 11 system is vula.uct.ac.za

These instructions assume that you can build Sakai 11 from source.

Obtain metatadata from ADFS

To start, you need the metadata for the ADFS installation. This may be provided by the ADFS administrator, or should be accessible via URL, e.g.

https://adfs.uct.ac.za/federationmetadata/2007-06/federationmetadata.xml

Save the metadata file in a location accessible by the Sakai tomcat process. In this example:

```
/data/sakai/otherdata/saml/federationmetadata.xml
```

Create keystore with keys for development and production system

To sign metadata and logout requests (which is required by ADFS), SAML requires a keystore with keys for the development and production system. Create these using the java keytool utility in the same location as the ADFS metadata:

```
cd /data/sakai/otherdata/saml/
```

```
keytool -genkeypair -alias vuladevkey -keypass changeit -keystore samlKeystore.jks -keyalg RSA
```

```
keytool -genkeypair -alias vulaprodkey -keypass changeit -keystore samlKeystore.jks -keyalg RSA
```

Notes:

a. It is essential that the key algorithm used here is RSA. SAML uses the same key algorithm used in these keys to sign the Logout Requests. These must be signed with RSA-SHA1, as ADFS does not accept DSA signatures.
b. The development and production systems must have different keys. ADFS will not allow two separate systems to use the same keys.

Configure container login in sakai.properties

For a configuration where users can login via SSO (ADFS) or internally, configure these settings in sakai.properties (replace “ADFS Login” with a name specific to your institution):

```
# don’t show the user id and password for login on the gateway site
top.login=false

# Enable SSO login
container.login=true
login.text=ADFS Login

# Second login link (bypasses container auth)
xlogin.enabled=true
xlogin.text=Guest Login
login.use.xlogin.to.relogin=false

# Enable the auth choice page. Only set this if container.login=true
login.auth.choice=true

# Set the icon or text you want for each. Generally you wouldn't use
both.
container.login.choice.text=ADFS Login
xlogin.choice.text = Guest Login

# SAML logout (ADFS) for account types that can authenticate via SSO
loggedOutUrl.staff=/sakai-login-tool/container/saml/logout
loggedOutUrl.student=/sakai-login-tool/container/saml/logout
```

Add UpnFilter for ADFS username recognition

Add this filter to your login source tree (to be contributed in https://jira.sakaiproject.org/browse/SAK-32065):

http://source.cet.uct.ac.za/svn/sakai/src_mods/trunk/login/login-tool/tool/src/java/org/sakaiproject/login/filter/UpnSamlFilter.java

This filter uses the attribute provided by ADFS with the identifier http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn as the Sakai user name (EID).

Configure login SAML XML files

1. Create two import file entries (one for the development and production system SAML configurations) in:

login/login-tool/tool/src/webapp/WEB-INF/applicationContext.xml

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

    <!-- Import both dev and production definitions using Spring Profiles -
->
    <import resource="xlogin-context.saml.adfs-prod.xml" />
    <import resource="xlogin-context.saml.adfs-dev.xml" />
```

```
</beans>
```

2. Then configure (in the same directory) `xlogin-context.saml.adfs-prod.xml` and `xlogin-context.saml.adfs-dev.xml` based on this sample http://source.cet.uct.ac.za/svn/sakai/src_mods/trunk/login/login-tool/tool/src/webapp/WEB-INF/xlogin-context.saml.devslscle001.xml

- In profile="devslscle001", replace devslscle001 with a short name for your production and development systems respectively. This is a Spring profile name which is used to load the correct SAML profile on startup.
- Replace all references to `https://devslscle001.uct.ac.za/` with the equivalent URL prefix for your development / production systems respectively.
- Replace all references to "vuladevkey" with the key names used in the keystore for your production and development systems respectively.
- Verify that the paths for `/data/sakai/otherdata/saml/federationmetadata.xml` and `file:/data/sakai/otherdata/saml/samlKeystore.jks` are correct.
- Replace the reference to adfs.uct.ac.za as the default IdP to your IdP's domain name here:

```
<property name="defaultIDP"
value="http://adfs.uct.ac.za/adfs/services/trust"/>
```

Build and run development system

Build the modified login module in the development system, and deploy Sakai.

Add the following to your JAVA_OPTS environment variable (typically in the Sakai start script):

-Dspring.profiles.active="devslscle001"

Replacing "devslscle001" with the short name for your development system configured above.

Obtain development system metadata

Once your development Sakai system has started, you can retrieve the auto-generated metadata like this:

wget https://devslscle001.uct.ac.za/sakai-login-tool/container/saml/metadata

(replacing devslscle001.uct.ac.za with your development system's FQDN).

With the exception of the certificate data (omitted here for clarity), your metadata should resemble this:

```
<?xml version="1.0" encoding="UTF-8"?>
<md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
entityID="https://devslscle001.uct.ac.za/saml" ID="https___devslscle001.uct.ac.za_saml">
  - <md:SPSSODescriptor protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"
    WantAssertionsSigned="true" AuthnRequestsSigned="false">
      - <md:KeyDescriptor use="signing">
          - <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
              - <ds:X509Data>
                    <ds:X509Certificate>(omitted)</ds:X509Certificate>
                </ds:X509Data>
            </ds:KeyInfo>
        </md:KeyDescriptor>
      - <md:KeyDescriptor use="encryption">
          - <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
              - <ds:X509Data>
                    <ds:X509Certificate>(omitted)</ds:X509Certificate>
                </ds:X509Data>
            </ds:KeyInfo>
        </md:KeyDescriptor>
        <md:SingleLogoutService Location="https://devslscle001.uct.ac.za/sakai-login-
            tool/container/saml/SingleLogout" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"/>
        <md:SingleLogoutService Location="https://devslscle001.uct.ac.za/sakai-login-
            tool/container/saml/SingleLogout" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"/>
        <md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</md:NameIDFormat>
        <md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</md:NameIDFormat>
        <md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</md:NameIDFormat>
        <md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</md:NameIDFormat>
        <md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName</md:NameIDFormat>
        <md:AssertionConsumerService Location="https://devslscle001.uct.ac.za/sakai-login-
            tool/container/saml/SSO" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" isDefault="true"
            index="0"/>
        <md:AssertionConsumerService Location="https://devslscle001.uct.ac.za/sakai-login-
            tool/container/saml/SSO" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Artifact" index="1"/>
    </md:SPSSODescriptor>
</md:EntityDescriptor>
```

Obtain production system metadata

Follow the same process as above, except alter the JAVA_OPTS on your development system to use the Spring profile name of your production system (e.g. -Dspring.profiles.active="vula").

Retrieve the auto-generated metadata from the development system URL path /sakai-login-tool/container/saml/metadata and save it as production metadata.

Configure ADFS

Provide the development system and production system metadata files to the ADFS Administrator, who should then import the metadata files and configure the ADFS trusts appropriately. See the screenshots in the related doc for examples.

ADFS must provide the UPN in the AuthnResponse to Sakai.

Test login with development system

Click on the “ADFS Login” button on your development system. The sequence of requests should be (abbreviated):

- Sakai GET /portal/login (302 response)
- Sakai GET /sakai-login-tool/container (200 response)
- ADFS POST /adfs/ls/
- Sakai POST /sakai-login-tool/container/saml/SSO (302 response)
- Sakai GET /sakai-login-tool/container/saml/container (302 response)
- Sakai GET /portal

Click on the Logout button on your development system. The sequence of requests should be:

- Sakai GET /portal/logout
- Sakai GET /sakai-login-tool/container/saml/logout
- ADFS GET /adfs/ls/?SAMLRequest=…
- Sakai GET /sakai-login-tool/container/saml/SingleLogout?SAMLResponse=… (302 response)
- Sakai GET /portal/

If you click on “ADFS Login” again, you should be prompted for crentials from ADFS, i.e. you should be properly logged out by the previous step.

Troubleshooting

To inspect the SAML requests and responses between systems, use Chrome Inspector (or similar in other browsers) to look at the POST and GET requests.

For POST requests, look at the SAMLRequest field in the POST form data. You can display the SAML XML using https://www.samltool.com/decode.php (as the form data is compressed).

For GET requests, look at the SAMLRequest URL parameter. To display the SAML, first URLDecode the string (e.g. https://urldecode.org/) then use https://www.samltool.com/decode.php

For Firefox, the https://addons.mozilla.org/en-US/firefox/addon/saml-tracer/ may be helpful.

The most likely issue is that the SAMLResponse after authentication (posted to /sakai-login-tool/container/saml/SSO) does not include a valid username. A valid response should include an AttributeStatement like this:

```
- <AttributeStatement>
  - <Attribute Name="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress">
      <AttributeValue>stephen.marquard@uct.ac.za</AttributeValue>
    </Attribute>
  - <Attribute Name="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn">
      <AttributeValue>01404877</AttributeValue>
    </Attribute>
</AttributeStatement>
```

where the UPN is the Sakai username (EID) that has been authenticated.