MUSE

MuseKnowledge™ Proxy and SAML Authentication

MuseGlobal, Inc.
One Embarcadero
Suite 500
San Francisco, CA 94111
415 896-6873
www.museglobal.com

MuseGlobal S.A
Calea Bucuresti
Bl. 27B, Sc. 1, Ap. 10
Craiova, România
40 251-413496
www.museglobal.ro

EduLib, S.R.L.
Calea Bucuresti
Bl. 27B, Sc. 1, Ap. 2
Craiova, România
40 351-420970
www.edulib.com

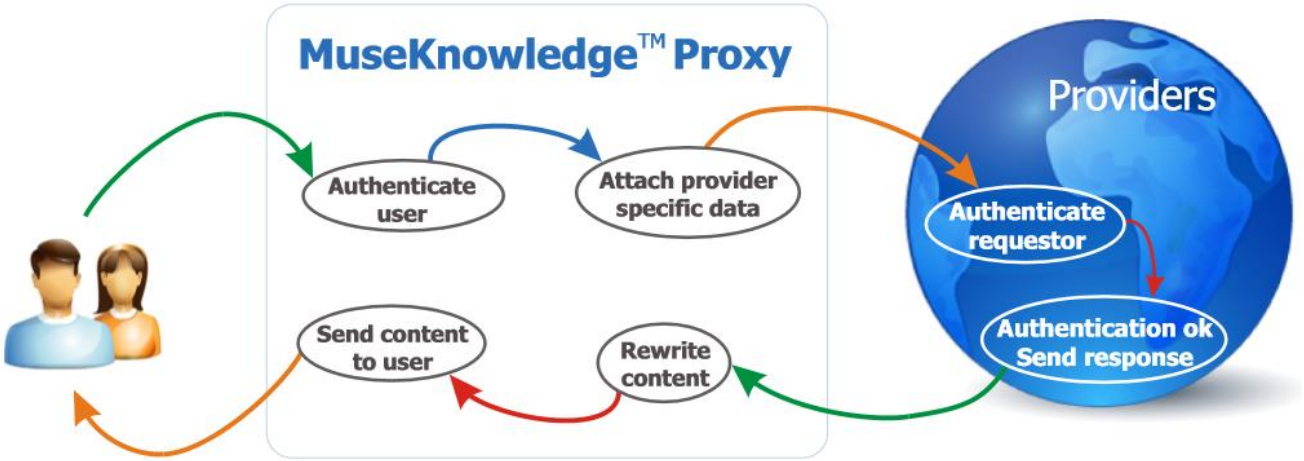
Version: 1.0
Date: 4th August 2016
Author: EduLib, S.R.L.

MuseKnowledge™ Proxy

Highly customizable multi-platform proxy server

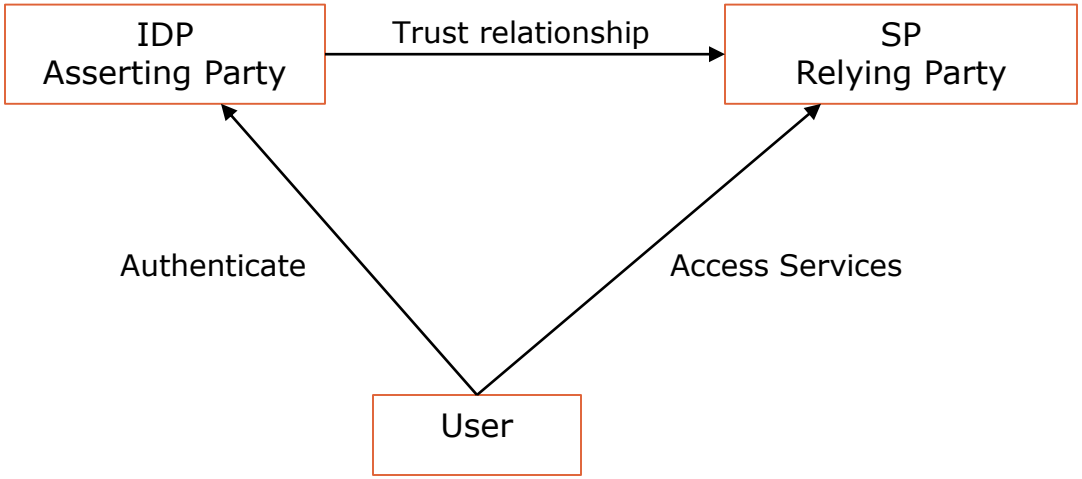
- Easy to use and configure via the MuseKnowledge™ Proxy Administrator Console;
- Gateway to authenticated restricted content;
- Rewriting web server;
- Web Access Management (WAM);
- Proxy server and reverse proxy;
- Supports SAML 2.0 as a Service Provider (SP).

More than 10 years within the Muse Federated Search Platform to manage the authentication to resources and the navigation to full text



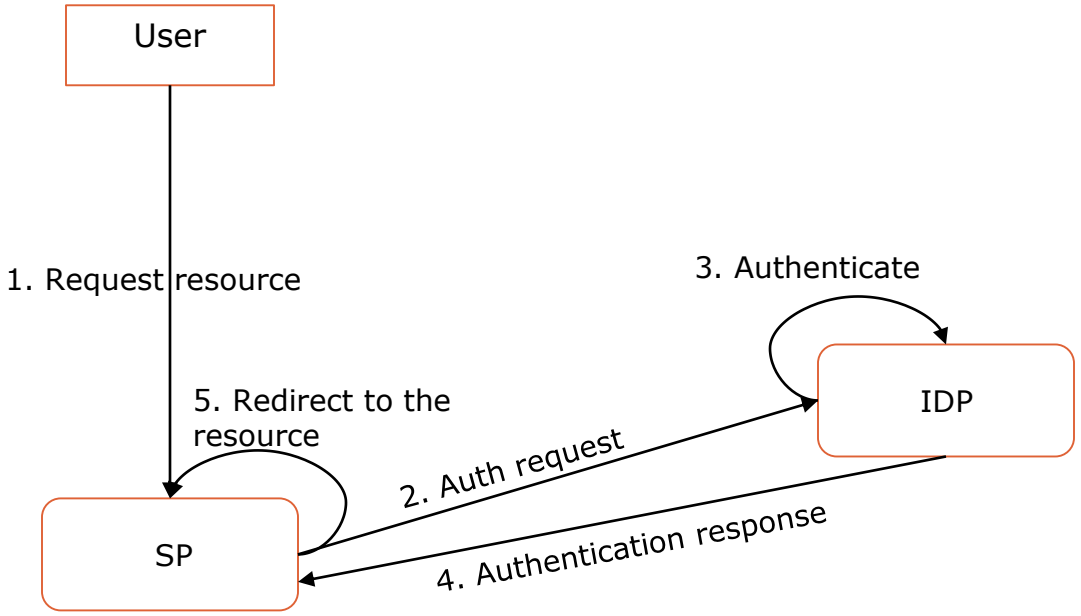
What is SAML?

- Security Assertion Markup Language;
- XML-based, open-standard data format for exchanging authentication and authorization data between parties, in particular, between an Identity Provider (IDP) and a Service Provider (SP);
- Single Sign-On;
- OASIS approved standard;
- Flexible and extensible protocol designed for integrations;



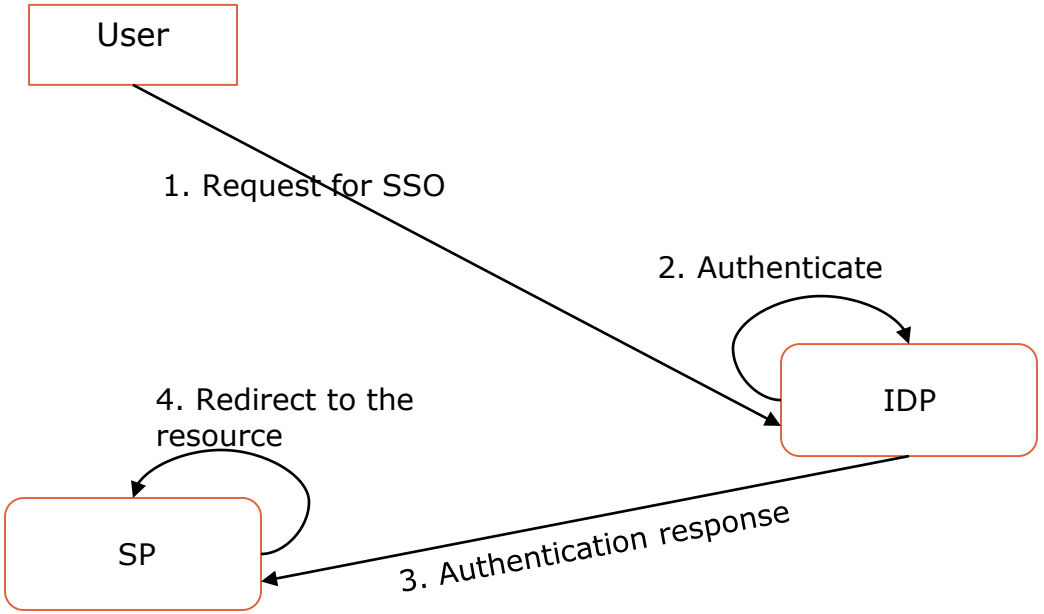
SP Initiated Single Sign-On

- End-user accesses directly a resource on the SP;



IDP Initiated Single Sign-On

- End-user accesses directly the IDP for SSO;
- Not anymore common and not the direction recommended; MuseKnowledge™ Proxy responds to IDP Initiated SSO (Unsolicited SSO) as long as the IDP allows for passing a RelayState, because the assertion point is different than the entry point in a MuseKnowledge™ Proxy application, or than a direct source link.



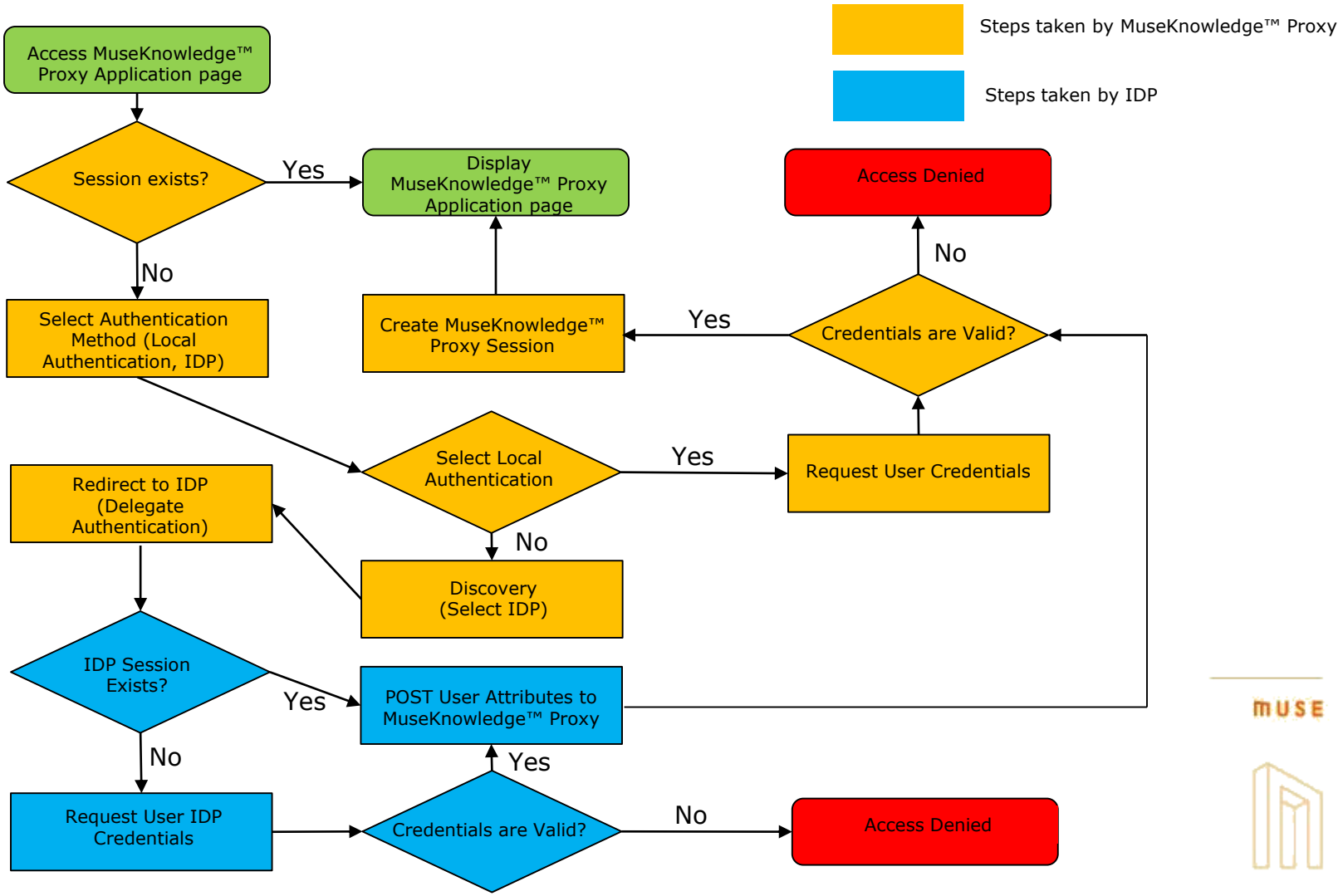
Features

- Theoretically all products supporting SAML 2.0 in Identity Provider mode should be compatible with Muse Proxy;
- Includes a local Discovery service;
- Supports external Discovery;
- Supports specifying the IDP metadata either by uploading the IDP metadata file or by specifying the IDP metadata URL with a local file backup with periodically refreshes;
- Supports specifying IDP metadata as a file/URL containing one *EntityDescriptor* or as multiple *EntityDescriptor* wrapped in *EntitiesDescriptor* (e.g. a federation) with filters eliminating conflicts if the SP metadata is also present in the same file;
- Easy restart of only the Servlet Engine, not affecting MuseKnowledge™ Proxy end-users already authenticated;
- Post-SAML authentication decisions via server side JavaScript on letting the user in the application, choosing a source group, choosing an attribute to be logged into the statistics;

Features

- Mapping SAML attributes which are then available to be used in the Muse Proxy HTML interface level.;
- Metadata management supporting adding IDP metadata and generation of SP metadata, pre-validation of IDP metadata to detect the need of certificates, tests for authentication, seeing SAML attributes, guidelines and more;
- The SAML authentication works combined with proxified widgets (forms), *?url=* like links;
- Other authentications, especially IP authentication (as a sufficient authentication) can be combined with SAML authentication.

MuseKnowledge™ Proxy SAML Authentication Scenario

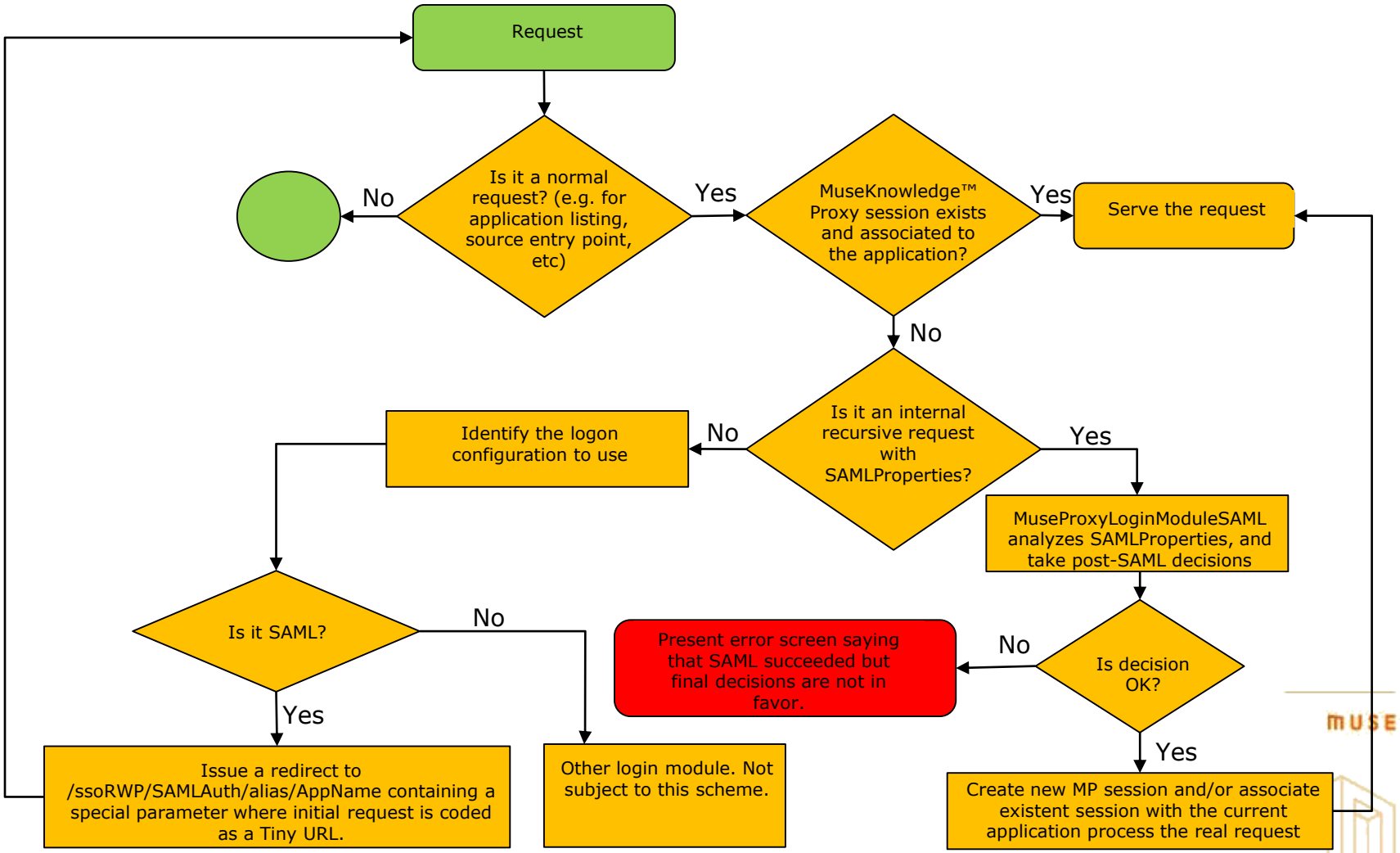


SAML Implementation in MuseKnowledge™ Proxy

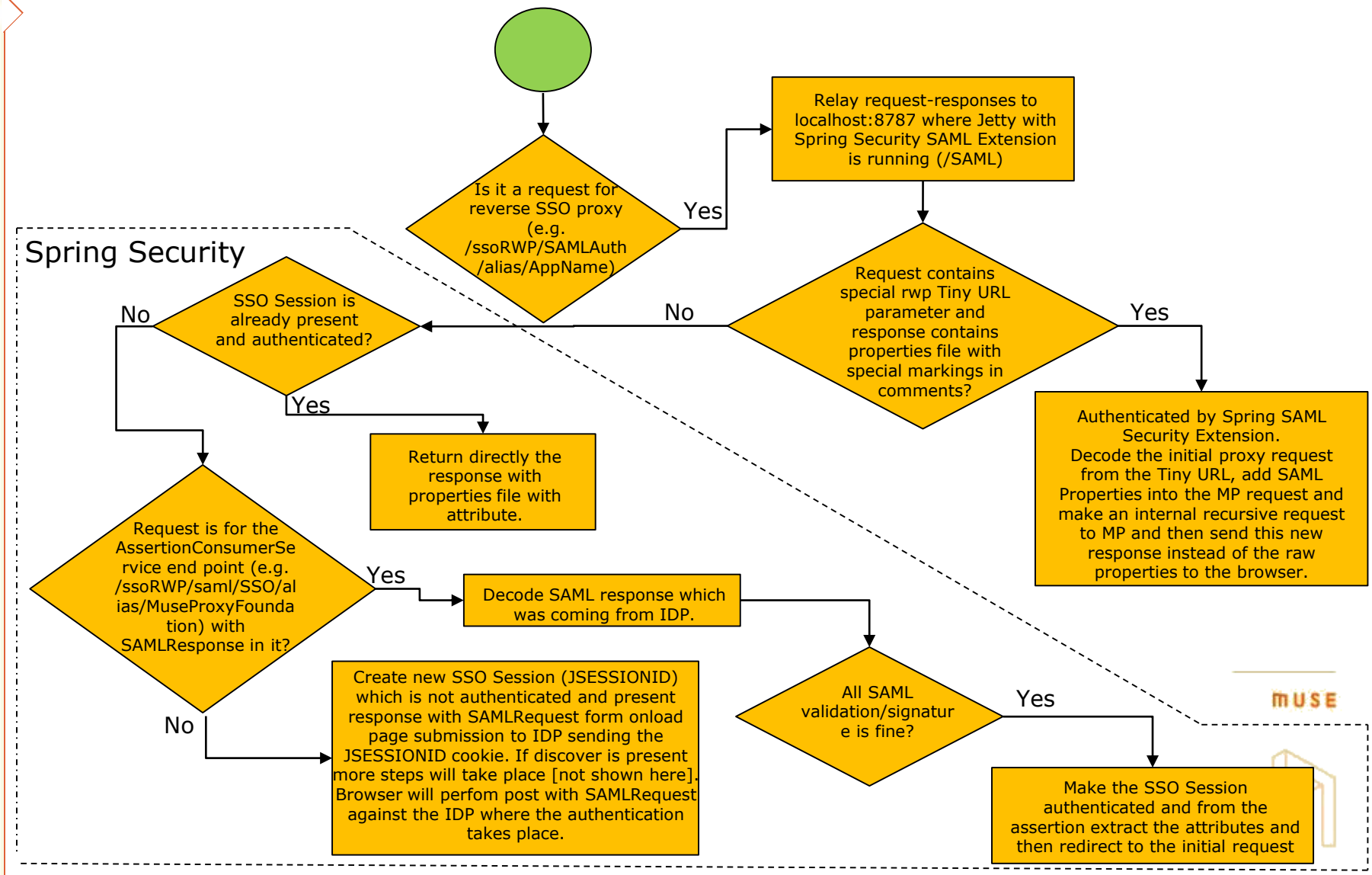
- Based on Spring Security SAML Extension and running inside Jetty Servlets Container which is integrated within MuseKnowledge™ Proxy;
- Supports ADFS, Okta, Shibboleth, OpenAM, Efecte EIM or Ping Federate;
- Indirect Authentication type as the authentication process happens on the Identity Provider (IDP) side; Because of this, the configuration is more complex than other login modules such as LDAP, IMAP, etc.;
- Multiple Service Provider for multiple customer applications (multi-tenancy);
- MuseKnowledge™ Proxy acts as a Reverse Proxy for certain paths and passes the request transparently to Spring Security SAML Extension inside Jetty; the URLs will contain /alias/AppIP, for example:

  http://proxy.museglobal.ro/ssoRWP/saml/SSO/alias/MuseProxyFoundation
- Jetty is embedded inside MuseKnowledge™ Proxy and controlled programmatically, hence only a single piece of software for handling SAML;
- All the communication MuseKnowledge™ Proxy <--> Jetty being done internally, only on localhost; from outside the host:port being the same either for a SAML Auth requests or for a usual proxy request.

MuseKnowledge™ Proxy SAML Workflow



MuseKnowledge™ Proxy SAML Workflow



SAML Configuration Requirements and Check List

Requirements:

- Make sure all server names are configured into *SERVER_NAMES*, including the load balanced name if applicable;
- MuseKnowledge™ Proxy administrator necessary skills: SAML, XML, SSL certificates.

Check List:

- Activate the internal Servlet Engine where SAML SP component is running (one time action);
- Identify the IDP(s) to use and configure references to its/theirs metadata;
- Generate and store the Service Provider metadata for the desired MuseKnowledge™ Proxy application;
- Publish the freshly SP generated metadata on the IDP(s) and if it is the case on the Discovery / Federation Services;
- MuseKnowledge™ Proxy application wiring and authentication decisions (e.g. sources group based on SAML attributes).

SAML Configuration Files and Locations

- Files and locations that have to do with SAML authentication:
  - *${MUSE_HOME}/proxy/webcontexts/ssoRWP/WEB-INF/classes/metadata/*

  is the storage for all the defined SP Metadata files as well as for the IDP ones if they are first downloaded and not referred through remote URLs.
  - *${MUSE_HOME}/proxy/webcontexts/ssoRWP/WEB-INF/securityContext-metadata.xml*

  Is an index for the IDP and all the alias SP metadata as well as an index for the signing and encryption keys used in the process.
  - *${MUSE_HOME}/proxy/webcontexts/ssoRWP/WEB-INF/classes/security/samlKeystore.jks*

  Is the storage for the certificates and private keys (according to Java terminology they are keypairs) used during SAML flows for signature and encryption.
  - *${MUSE_HOME}/proxy/webcontexts/Applications/${APP_ID}/profiles/AuthenticationGroups.xml*

  MuseKnowledge™ Proxy Application authentication file.
  - *${MUSE_HOME}/proxy/webcontexts/Applications/${APP_ID}/profiles/login/ProxyLoginModuleSAML.xml*

  MuseKnowledge™ Proxy SAML login module authentication file.

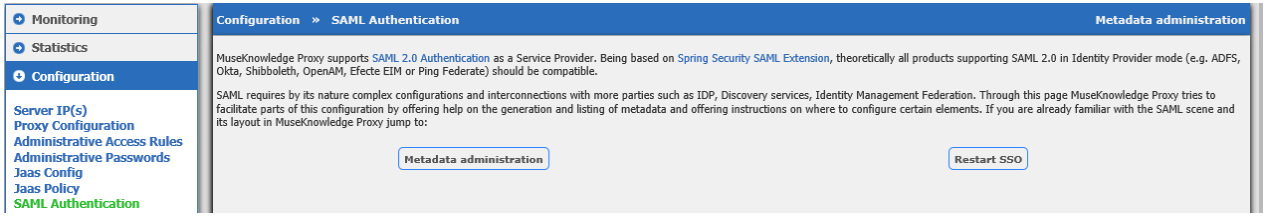
Activate the Internal Servlet Engine

- Edit the file:

  ${MUSE_HOME}/proxy/MuseProxy.xml

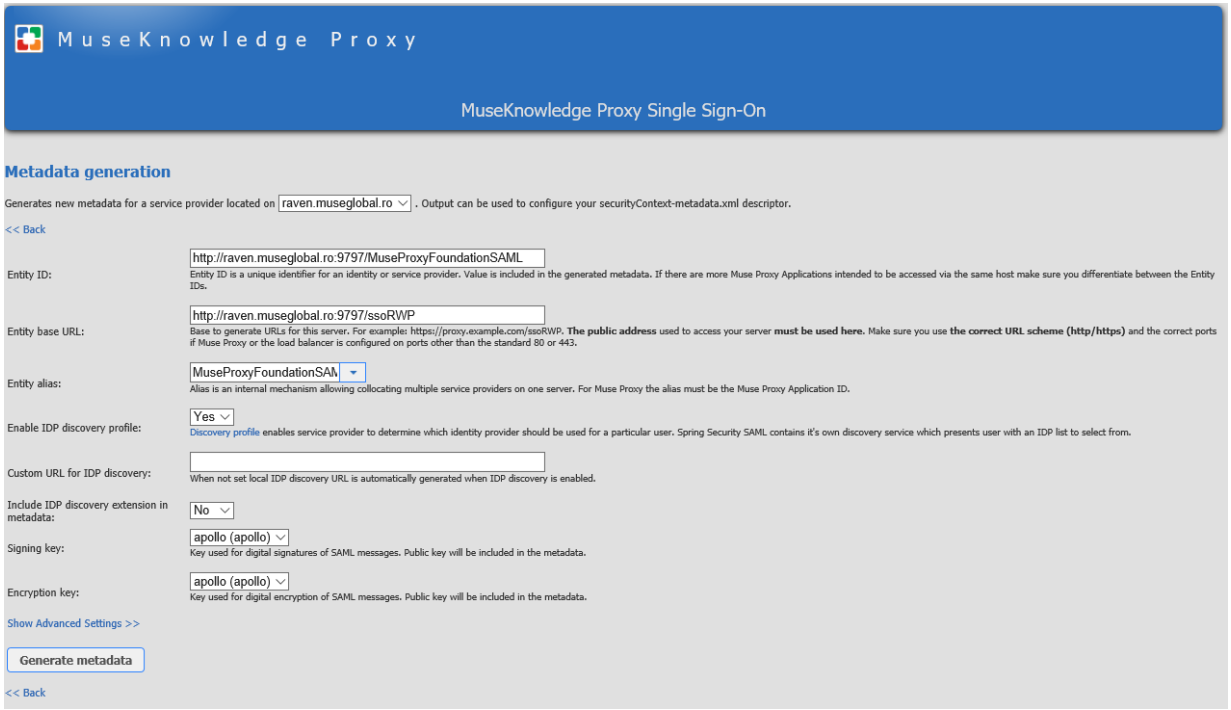
  and set to *true* the value in the *SERVLET_ENGINE_ENABLED* node;
- Restart the MuseKnowledge™ Proxy service;
- Jetty Servlet Engine configuration file:

  ${MUSE_HOME}/proxy/ServletEngine.xml
- Accessible only on localhost;
- Restart feature from the MuseKnowledge™ Proxy Administrator Console, *Configuration –> SAML Authentication* left menu, *Restart SSO* button:



Generate new Service Provider Metadata

- MuseKnowledge™ Proxy Administrator Console, *Configuration -> SAML Authentication*, *Metadata Administration* button, *Generate new service provider metadata* button;
- Mandatory to configure *SERVER_NAMES* option in the *${MUSE_HOME}/proxy/MuseProxy.xml* file to reflect all the fully qualified domain name used to access MuseKnowledge™ Proxy.



Generate new Service Provider Metadata

- Click on *Create File* button to save the metadata under folder:

${MUSE_HOME}/proxy/webcontexts/ssoRWP/WEB-INF/classes/metadata

- Update the identity provider(s) with the generated metadata;
- Modify bean "metadata" in

${MUSE_HOME}/proxy/webcontexts/ssoRWP/WEB-INF/securityContext-metadata.xml

and include the generated content from the Configuration section under the <*list*> element.

- Restart the Servlet Engine from the MuseKnowledge™ Proxy Administration console, *Configuration -> SAML Authentication*, *Restart SSO* button;
- In case a single IDP is used for authentication for the application add its entity ID in *${MUSE_HOME}/proxy/webcontexts/Applications/MuseProxyFoundationSAML/profiles/login/ProxyLoginModuleSAML.xml*

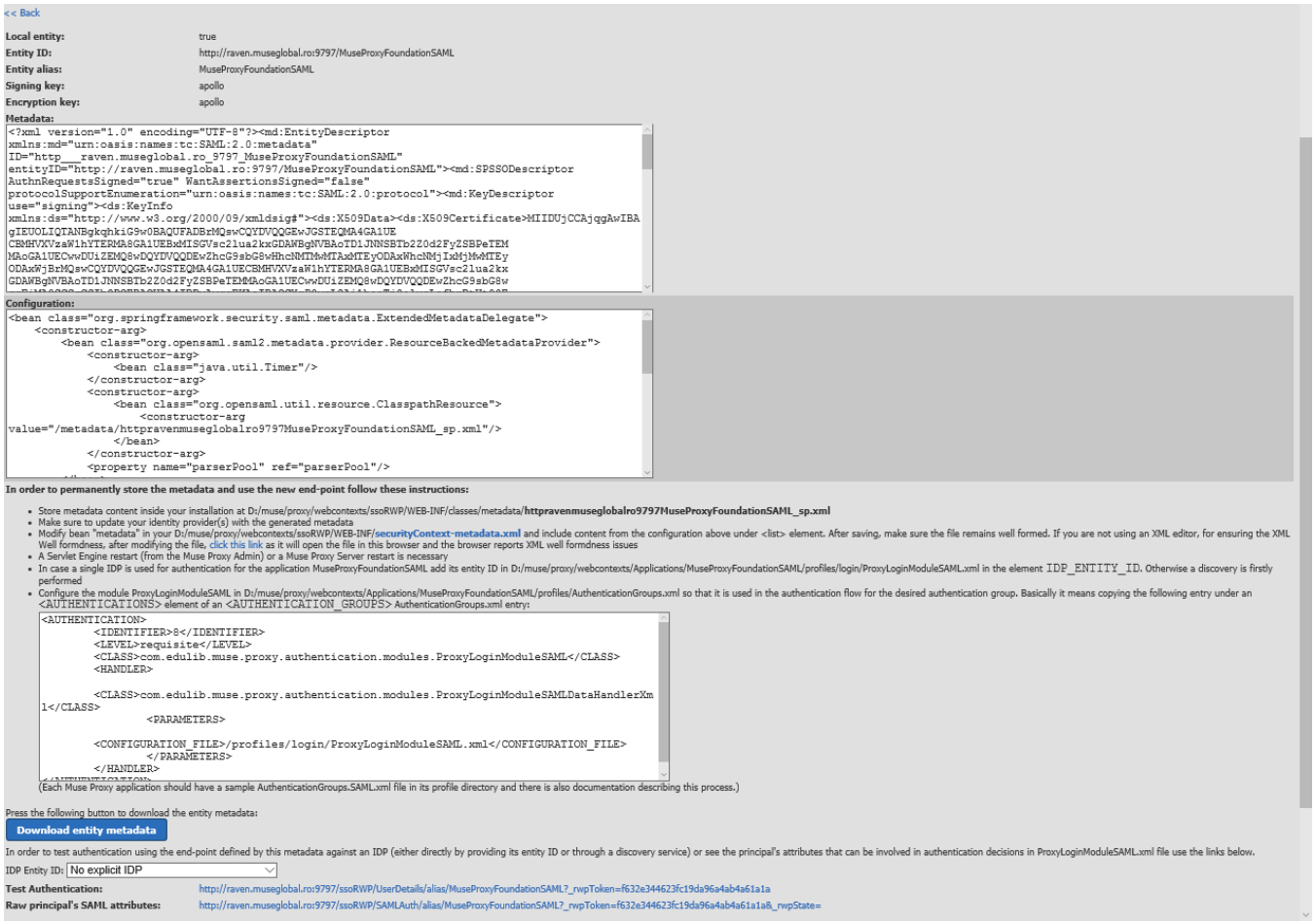
in the element *IDP_ENTITY_ID*. Otherwise a discovery is firstly performed;

- Configure the module ProxyLoginModuleSAML in *${MUSE_HOME}/proxy/webcontexts/Applications/MuseProxyFoundationSAML/profiles/AuthenticationGroups.xml*

so that it is used in the authentication flow for the desired authentication group.

Generate new Service Provider Metadata

- Metadata generation and detailed configuration steps:



Generate new Identity Provider Metadata

- MuseKnowledge™ Proxy Administrator Console, *Configuration -> SAML Authentication*, *Metadata Administration* button, *Add new identity provider metadata* button;
- 2 methods available:
  - *By URL*. Provide the URL pointing to the IDP metadata (it can be a federation URL, too);
  - *By Upload*.
- Modify bean "metadata" in

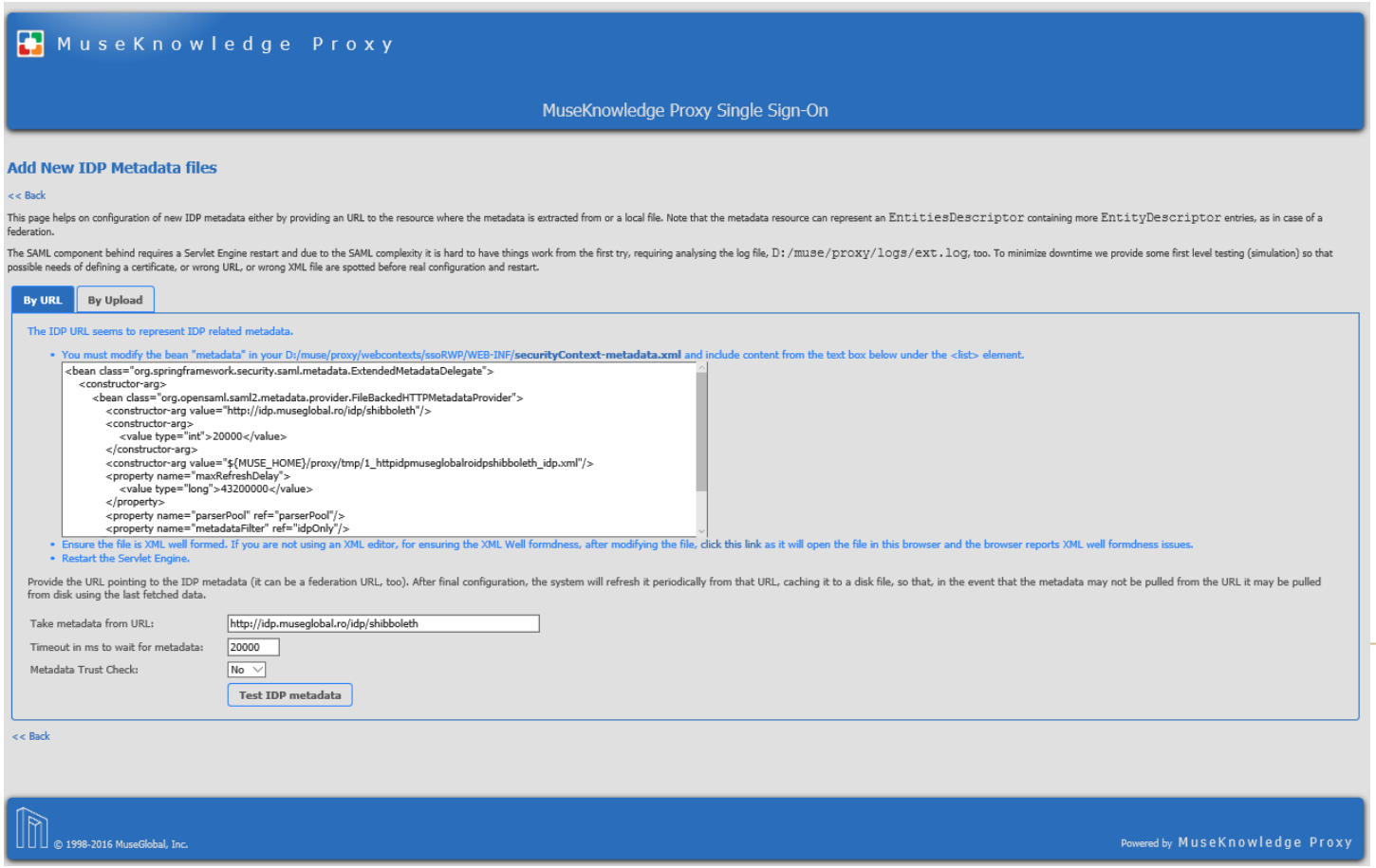
${MUSE_HOME}/proxy/webcontexts/ssoRWP/WEB-INF/securityContext-metadata.xml

and add the generated content under the <*list*> element.

- Restart the Servlet Engine from the MuseKnowledge™ Proxy Administration console, *Configuration -> SAML Authentication*, *Restart SSO* button;

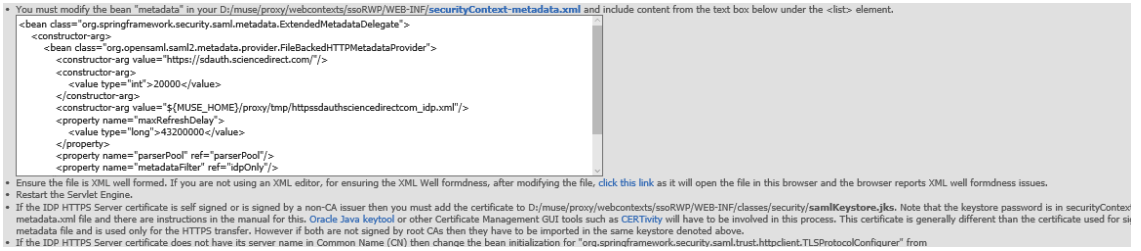
Generate new Identity Provider Metadata

- Add IDP Metadata and complete steps.

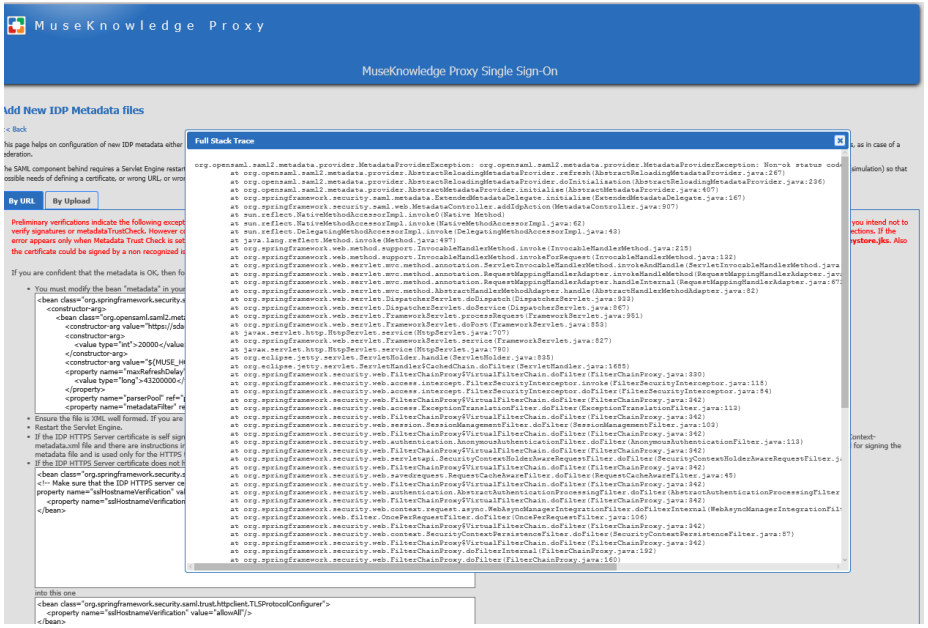


Configuration Helpers

- Various helper tools are available to easily spot the errors:
  - XML file view inside the browser to detect not-wellformed XML files;



  - See full stack trace of errors.



ProxyLoginModuleSAML.xml Configuration File and Final Authentication Decisions

- Make sure that the ProxyLoginModuleSAML.xml file is linked to from the AuthenticationGroups.xml file;
- Use the *IDP_ENTITY_ID* element and fill it with an entity ID to avoid discovery screens;
- *INPUT* XML element:
  - defines input parameters to be used as global JavaScript variables in the *SCRIPT* section by mapping between a variable id and the friendly name from Principal's SAML Attributes;

```
<INPUT>
  <!-- The 'id' attribute of PARAM is the JavaScript variabile and source is the friendly name from the
  Principal's SAML attributes. If a friendly name is not available then the name (urn:...) has to be used. Note
  that an attribute has both a friendly name and a name, then it can be accessed using its friendly name only.
  -->
  <PARAM id="idp" source="issuer"/>

  <PARAM id="eduPersonPrincipalName" source="eduPersonPrincipalName"/>
  <PARAM id="eduPersonAffiliation" source="eduPersonAffiliation"/>
  <PARAM id="eduPersonTargetedID" source="eduPersonTargetedID"/>

  <PARAM id="displayName" source="displayName"/>
  <PARAM id="email" source="mail"/>

  <PARAM id="uid" source="uid"/>
  <PARAM id="o" source="o"/>
```

```
<saml2:Attribute FriendlyName="eduPersonPrincipalName"
   Name="urn:oid:1.3.6.1.4.1.5923.1.1.1.6"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
   <saml2:AttributeValue>gabi@sharepoint.museglobal.com</saml2:AttributeValue>
</saml2:Attribute>
<saml2:Attribute FriendlyName="displayName" Name="urn:oid:2.16.840.1.113730.3.1.241"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
   <saml2:AttributeValue>Gabriel Toma-Tumbar</saml2:AttributeValue>
</saml2:Attribute>
```

ProxyLoginModuleSAML.xml Configuration File and Final Authentication Decisions

- *OUTPUT* XML element:
  - defines the output parameters usable in the Freemarker templates in the Muse Proxy Application interface;
  - contains *PARAM* elements with *id* and *source* attribute;
  - the *id* represents the name of the property to be further used in the Muse Proxy application interface;

```
<OUTPUT>
  <!-- List of parameters computed by script sent back to Muse Proxy.
   id  = the login parameter name sent back by script to Muse Proxy.
  The parameters can then be used in the Freemarker templates in the Muse Proxy Application interface.

  source = the javascript variable name. If it missing then source will have the same value as id. -->
  <PARAM id="idp" source="idp"/>
  <PARAM id="userID" source="userID"/>
  <PARAM id="eduPersonPrincipalName" source="eduPersonPrincipalName"/>
  <PARAM id="o" source="o"/>
  <PARAM id="displayName" source="displayName"/>
</OUTPUT>
```

ProxyLoginModuleSAML.xml Configuration File and Final Authentication Decisions

- *SCRIPT* XML element (optional):
  - contains under a CDATA construction the JavaScript code;
  - the variables defined in the INPUT section can be used for comparisons for final decision on the authentication;
  - the following global variables have to be used:
    - *succeeded* variable say if authentication pass or not;
    - *logUserID* variable which caries the value of the user identifier to be logged;
    - *sourcesGroupID* selects the source group to be used. If missing or empty, the default group is implied;
    - *errorMessage* variable that store the error message sent back to login module.

ProxyLoginModuleSAML.xml Configuration File and Final Authentication Decisions

```
<![CDATA[
var succeeded = false;
var sourcesGroupID = null;
var errorMessage = null;
var logUserID = null;

doLogin();
function doLogin() {
  if (typeof idp === 'undefined' || idp === null || idp === '') {
    // idp is undefined or null.
    succeeded = false;
    errorMessage = 'IDP is undefined';
    return false;
  }
       // Use below block instead of the next else statement if you want to ensure that only certain IDPs can access this application (in case a discovery
service is used, for example)
  /*
  else if (idp !== 'http://idp.ssocircle.com') {
      succeeded = false;
       errorMessage = 'This IDP is not allowed for this application.';
       return false;
  } else {
    succeeded = true;
  } */
   else {
        succeeded = true;
        // Use code as the following one to assign a certain source group for a certain condition. Use contains() instead of === to cope with multi-valued
attributes.
        // Make sure that the source group ID are defined as by default the application contains only one source group.
        /*
        if (eduPersonAffiliation.contains("teacher")) {
                sourcesGroupID = "1";
        } else if (eduPersonAffiliation.contains("student")) {
                sourcesGroupID = "2";
        } else {
                sourcesGroupID = "3";
                        }*/
  }
  logUserID = eduPersonPrincipalName;
}
]]>
```

References

- Muse Proxy.pdf;
- Muse Proxy Advanced Configuration.pdf;
- Muse Proxy Administrator Console.pdf;

MUSE

MuseKnowledge™ Proxy and SAML Authentication