KnoxSSO and Okta

SAML v2 for Hadoop Web Applications

Introduction

Apache Knox with KnoxSSO + pac4j provider enables the use of a number of new authentication and SSO solutions for accessing and developing KnoxSSO enabled applications including, Ambari, Ranger, Hadoop UIs and custom built applications that utilize REST APIs through Knox. These capabilities will be available in the Knox 0.8.0 release.

This paper illustrates the integration of the Okta identity service offering by leveraging the pac4j provider SAML capabilities in Apache Knox. The same sort of flow that is described below would be available for Ambari and Ranger or any KnoxSSO participating application.

Okta

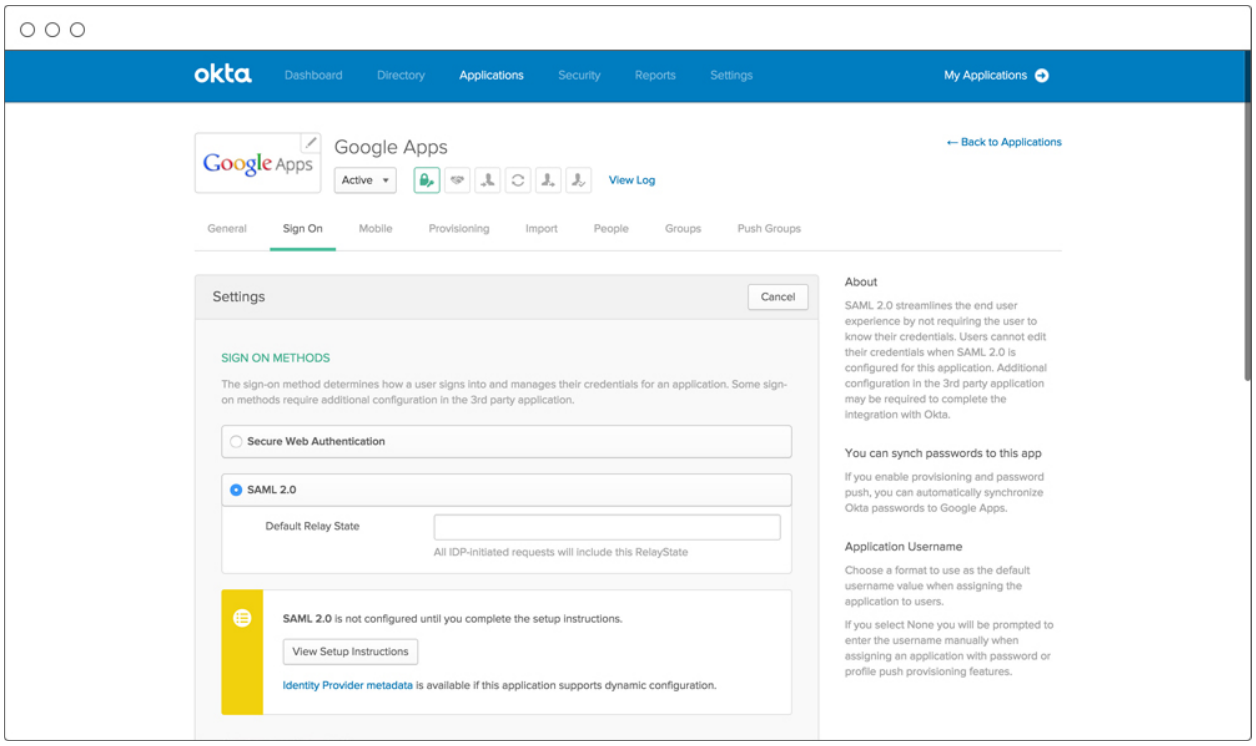
From their site (https://www.okta.com/):

Authentication you can trust

Behind the scenes, Okta enables SSO in one of two ways:

SAML

Security Assertion Markup Language (SAML) is a trusted format for exchanging authentication data. Okta enables single sign-on into SAML-enabled apps by brokering the information transfer between users and service providers.



SIGN ON METHODS

The sign-on method determines how a user signs into and manages their credentials for an application. Some sign-on methods require additional configuration in the 3rd party application.

○ Secure Web Authentication

● SAML 2.0

Default Relay State

All IDP-initiated requests will include this RelayState

SAML 2.0 is not configured until you complete the setup instructions.

Identity Provider metadata is available if this application supports dynamic configuration.

About

SAML 2.0 streamlines the end user experience by not requiring the user to know their credentials. Users cannot edit their credentials when SAML 2.0 is configured for this application. Additional configuration in the 3rd party application may be required to complete the integration with Okta.

You can synch passwords to this app

If you enable provisioning and password push, you can automatically synchronize Okta passwords to Google Apps.

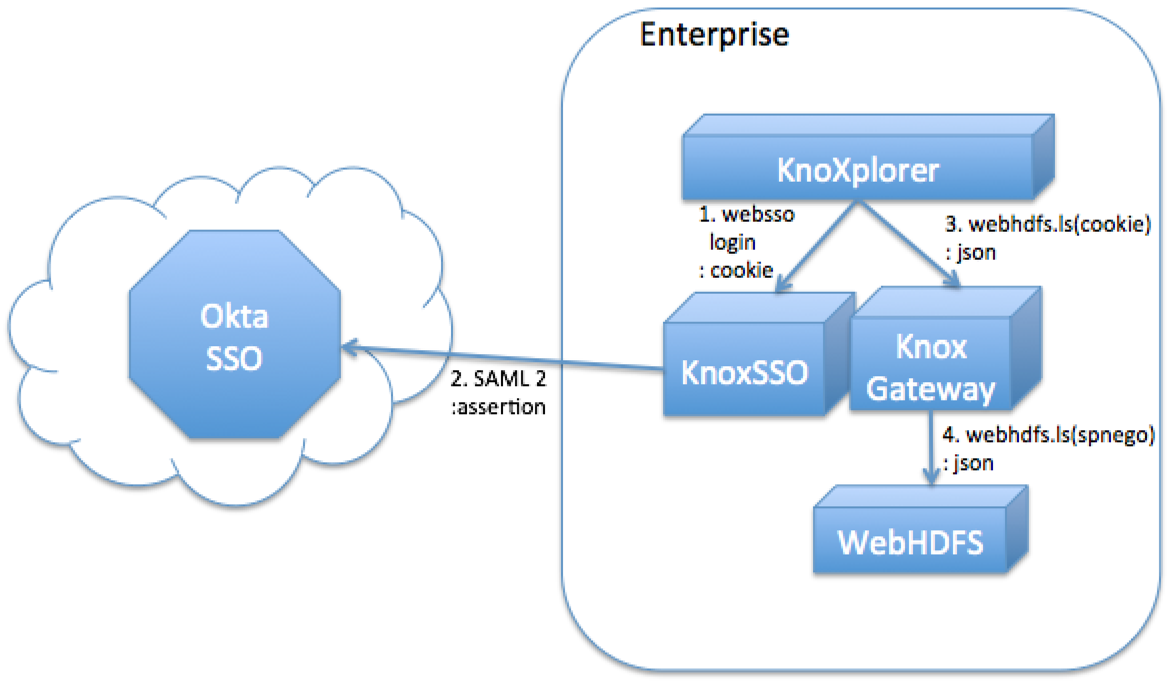
Application Username

Choose a format to use as the default username value when assigning the application to users.

If you select None you will be prompted to enter the username manually when assigning an application with password or profile push provisioning features.

Tutorial

This paper illustrates the use of Okta's SSO with SAML option for a simple KnoxSSO participating application called KnoXplorer. The following figure shows how KnoXplorer only ever needs to be aware of KnoxSSO. The underlying authentication mechanism is isolated from the participating application. In this case, KnoxSSO negotiates an authentication with Okta via the SAML 2 protocol.



Prerequisites

1. Create a /etc/hosts entry that maps www.local.com to 127.0.0.1. This is required because HTTP Cookies are used and this example is setup to work with this particular host/domain. Add a line like this to your /etc/hosts file.
   127.0.0.1 www.local.com
2. Clone a copy of the KnoXplorer project:
   git clone https://github.com/lmccay/knoxplorer.git
3. Deploy the knoxsso.xml topology file from the doc into your local knox instance. You can do this by copying the contents of the sample knoxsso.xml in this document into a new <GATEWAY_HOME>/conf/topologies/knoxsso.xml file.
4. Deploy the sandbox.xml topology file from the doc into your local knox instance. You can do this by copying the contents of the sample sandbox.xml in this document over the existing content of the <GATEWAY_HOME>/conf/topologies/sandbox.xml file.
5. Run KnoXplorer with the command below:
   start.py

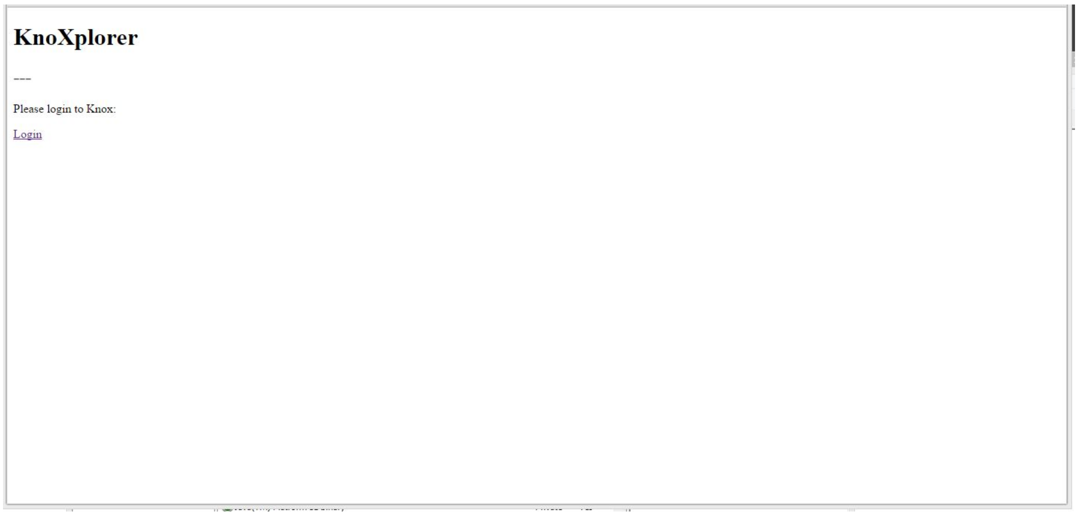
KnoXplorer (https://github.com/lmccay/knoxplorer)

To demonstrate the integration between KnoxSSO and Okta for new application development the following KnoXplorer application will be used. This is a simple demo of consuming Hadoop REST APIs from javascript in a web browser through Apache Knox using the SSOCookie provider and CORS capabilities in Knox.

Once logged in through KnoxSSO the resulting hadoop-jwt cookie is used to request LISTSTATUS calls to WebHDFS through Knox. The KnoXplorer application only knows that it is relying on KnoxSSO and nothing about the underlying SSO provider (in this case Okta).

Test Integration with Okta

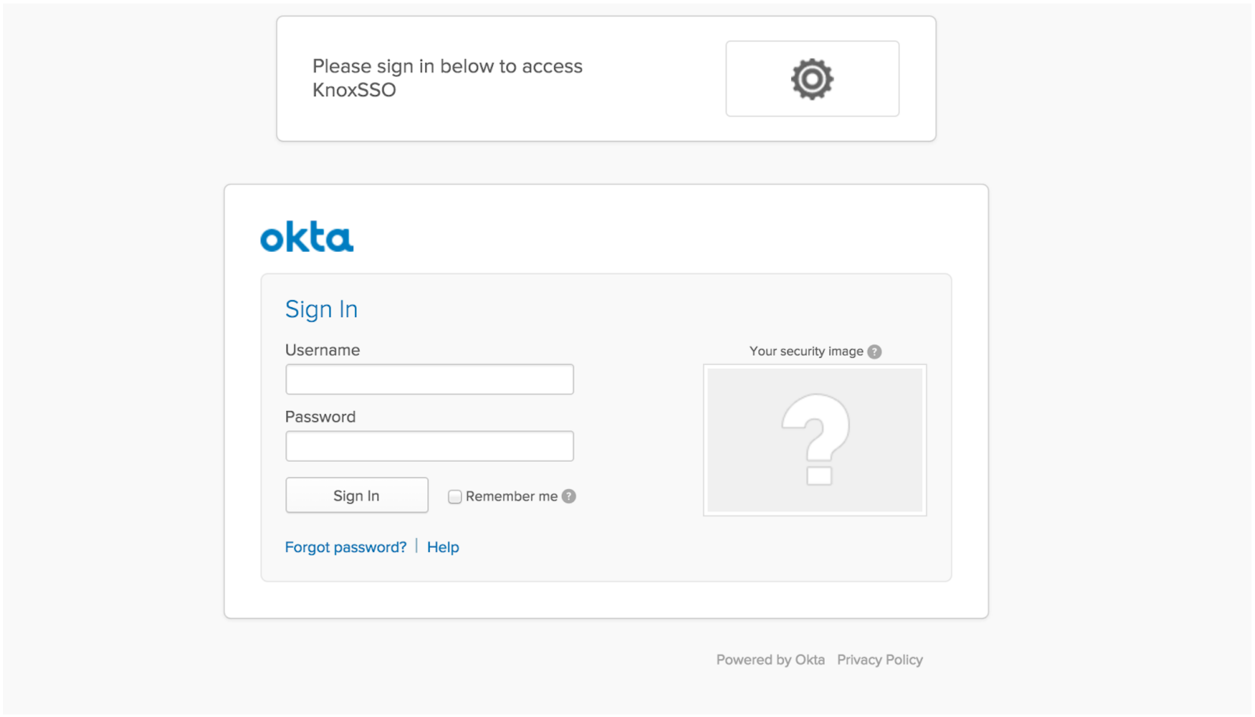
1. Open KnoXplorer in a broser at http://www.local.com:8000


KnoXplorer

Please login to Knox:

Login

2. This will result in KnoXplorer receiving an error when making REST calls so it presents the user with a login page. This link makes a request to the KnoxSSO service to authenticate which results in a redirect to the Okta server.


Please sign in below to access KnoxSSO

okta

Sign In

Username

Password

Sign In Remember me

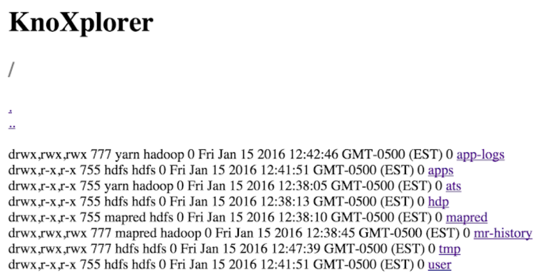
Your security image

Forgot password? | Help

Powered by Okta Privacy Policy

3. When presented with a login form, fill it out with these credentials (guest/Gu3stp@assword) and submit it to the Okta server. This will result in a SAML protocol POST to the callback URL for KnoxSSO. The SAML assertion will be processed via the pac4j provider and the authenticated identity normalized into a Java Subject. The successful authentication continues the processing through the provider's chain and the identity assertion provider must use appropriate principal mapping to establish the effective username. The effective username is what the KnoxSSO service will put into the JWT token to be presented as a Cookie to all participating applications.

4. After a brief signing in page you should be redirected to the original KnoXplorer page. If you are interested you may find the hadoop-jwt cookie using Chrome's developer tools - if the cookie is configured to not be secure only. It should be a session cookie set as HttpOnly and Secure. The service parameter knoxsso.cookie.secure.only for the KnoxSSO service in the knoxsso.xml topology controls the secure only setting of the cookie. If you don't see the cookie in the developer tools you may need to adjust this setting.

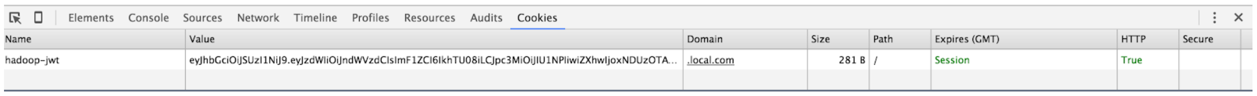


KnoXplorer

/

.
..

```
drwx,rwx,rwx 777 yarn hadoop 0 Fri Jan 15 2016 12:42:46 GMT-0500 (EST) 0 app-logs
drwx,r-x,r-x 755 hdfs hdfs 0 Fri Jan 15 2016 12:41:51 GMT-0500 (EST) 0 apps
drwx,r-x,r-x 755 yarn hadoop 0 Fri Jan 15 2016 12:38:05 GMT-0500 (EST) 0 ats
drwx,r-x,r-x 755 hdfs hdfs 0 Fri Jan 15 2016 12:38:13 GMT-0500 (EST) 0 hdp
drwx,r-x,r-x 755 mapred hdfs 0 Fri Jan 15 2016 12:38:10 GMT-0500 (EST) 0 mapred
drwx,rwx,rwx 777 mapred hadoop 0 Fri Jan 15 2016 12:38:45 GMT-0500 (EST) 0 mr-history
drwx,rwx,rwx 777 hdfs hdfs 0 Fri Jan 15 2016 12:47:39 GMT-0500 (EST) 0 tmp
drwx,r-x,r-x 755 hdfs hdfs 0 Fri Jan 15 2016 12:41:51 GMT-0500 (EST) 0 user
```



Name	Value	Domain	Size	Path	Expires (GMT)	HTTP	Secure
hadoop-jwt	eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJndWVzdCIsImF1ZCI6IkhTU08iLCJpc3MiOiJIU1NPIiwiZXhwIjoxNDUzOTA...	.local.com	281 B	/	Session	True	

Topologies

The contents of these topology files can be copied into your {GATEWAY_HOME}/conf/topologies directory.

knoxsso.xml

The knoxsso.xml topology describes the manner in which a client acquires a KnoxSSO websso cookie/token. The pac4j federation provider allows the integration of a number of authentication solutions. In this case, the openid connect capability is being leveraged to integration the cloud based Privakey identity service.

Please take note of the need to encode the ampersand within the saml.serviceProviderEntityId parameter as "&amp;" as well as the need to include a value for the saml.serviceProviderMetadataPath - the file location here doesn't need to exist. There is a bug that will throw a NPE if saml. serviceProviderMetadataPath is not included even though the actual metadata will be served up to the IdP via request.

```
<topology>

   <gateway>
```

```
  <provider>
      <role>federation</role>
      <name>pac4j</name>
      <enabled>true</enabled>
      <param>
        <name>pac4j.callbackUrl</name>
   <value>https://www.local.com:8443/gateway/knoxsso/api/v1/websso</value>
      </param>

      <param>
        <name>clientName</name>
        <value>SAML2Client</value>
      </param>

      <param>
        <name>saml.identityProviderMetadataPath</name>
        <value>https://dev-122415.oktapreview.com/app/exk5nc5z1xbFKb7nH0h7/sso/saml/metadata</value>
      </param>

      <param>

        <name>saml.serviceProviderMetadataPath</name>
        <value>/tmp/sp-metadata.xml</value>
      </param>

      <param>
        <name>saml.serviceProviderEntityId</name>
        <value>https://www.local.com:8443/gateway/knoxsso/api/v1/websso?pac4jCallback=true&amp;client_name=SAML2Client</value>
      </param>
  </provider>
  <provider>
      <role>identity-assertion</role>
      <name>Default</name>
      <enabled>true</enabled>
      <param>
        <name>principal.mapping</name>
        <value>guest@example.com=guest;</value>
      </param>
  </provider>
</gateway>
```

```
  <service>

      <role>KNOXSSO</role>

      <param>

        <name>knoxsso.cookie.secure.only</name>

        <value>true</value>

     </param>

     <param>

       <name>knoxsso.token.ttl</name>

       <value>100000</value>

     </param>

     <param>

        <name>knoxsso.redirect.whitelist.regex</name>

        <value>^https?:\/\/(www\.local\.com|localhost|127\.0\.0\.1|0:0:0:0:0:0:0:1|::1):[0-9].*$</value>

     </param>

  </service>

</topology>
```

sandbox.xml

The sandbox.xml topology is the descriptor file that defines the access to the Hadoop cluster itself - the services exposed, the providers used to protect the services, etc. For the usecase of protecting Hadoop resources with KnoxSSO authentication, we need to define the need to present an SSO cookie by using the SSOCookieProvider federation provider. We also need to enable Cross Origin Resource Sharing (CORS). This is done via the WebAppSec provider and the cors.enabled parameter.

- Note that the location of each service within the <url> markup needs to be tailored to the environment. These have been marked red to help identify them.

```
<?xml version="1.0"?>

<topology>

 <gateway>

  <provider>

    <role>webappsec</role>

    <name>WebAppSec</name>

    <enabled>true</enabled>

    <param>

      <name>cors.enabled</name>

      <value>true</value>

    </param>

  </provider>

  <provider>

    <role>federation</role>

    <name>SSOCookieProvider</name>

    <enabled>true</enabled>

    <param>
```

```
      <name>sso.authentication.provider.url</name>
      <value>https://www.local.com:8443/gateway/knoxsso/api/v1/websso</value>
    </param>
  </provider>
  <provider>
    <role>identity-assertion</role>
    <name>Default</name>
    <enabled>true</enabled>
  </provider>
</gateway>

<service>
  <role>NAMENODE</role>
  <url>hdfs://localhost:8020</url>
</service>
<service>
  <role>JOBTRACKER</role>
  <url>rpc://localhost:8050</url>
</service>
<service>
  <role>WEBHDFS</role>
  <url>http://localhost:50070/webhdfs</url>
</service>
<service>
  <role>WEBHCAT</role>
  <url>http://localhost:50111/templeton</url>
</service>
<service>
  <role>OOZIE</role>
  <url>http://localhost:11000/oozie</url>
</service>
<service>
  <role>WEBHBASE</role>
  <url>http://localhost:60080</url>
</service>
<service>
  <role>HIVE</role>
  <url>http://localhost:10001/cliservice</url>
</service>
<service>
```

```
   <role>RESOURCEMANAGER</role>
   <url>http://localhost:8088/ws</url>
 </service>
</topology>
```