WaTTS RCauth plugin - Obtaining IOTA proxy certificates with OIDC

Introduction

WaTTS (the INDIGO Token Translation Service) is a plugin-based Token Translation Service. Enabled by plugins, translation to various credentials (e.g. SSH keys, X.509, S3 access tokens, etc.) are supported. WaTTS accepts many OIDC providers (EGI-CheckIn, B2Access, Human Brain, Google and Indigo IAM).

In this pilot, we have implemented a plugin to demonstrate the integration with the RCauth.eu online CA. The WaTTS plugin implements a MasterPortal and Vo-Portal for the RCauth CA. After authenticating in WaTTS with any OIDC Provider, a certificate may be requested. For this, the user is redirected to the RCauth CA WAYF. There, she may choose which home-IdP to use for authenticating to RCauth. In this authentication step, the LoA must be high enough to fulfill policy requirements. To date B2Access, Checkin and a set of individual IdPs support these requirements.

NOTE:

In the first version of this pilot, we will use the development instance of RCauth, which does not use a WAYF but redirects users directly to the EGI-AAI-Dev instance.

Upon successful authentication at RCauth.eu, the users browser is redirected back to WaTTS, where he is given a proxy of his certificate. The full certificate is stored inside a myProxy secure credential store. This store is not part of WaTTS, but a separate service (also shown in the picture below). Development to provide integration with VOMS, i.e. a VOMS proxy, is on-going.

Commandline / API Access:

One goal of this pilot is to show the command line (CLI) access. Once the user has his OIDC-Access token on the command line, he can use the REST interface of WaTTS (we encourage the appropriate client wattson (https://github.com/indigo-dc/wattson) to retrieve his grid proxy.

Additional option is for the user to upload its SSH public key and then the key to obtain the proxy certificate via command line. For this, a separate plugin is provided.

Detailed description

Goal of the pilot is to demonstrate how WaTTS can be leveraged to deliver X.509 certificates of RCauth. In addition, support for usage from the commandline is demonstrated.

Pilot example is following:

User first needs to authenticate itself at WaTTS using one of the available OIDC providers. So far, EGI CheckIn service, EUDAT, Human Brain Project, INDIGO Access Management, and Google are supported. More OIDC providers can be provided in the future. After the successful login, a number of available plugins is displayed to the user. For this demo, X509_RCauth_Pilot is the plugin used to provide the service, and user request a proxy certificate by clicking on the Request button. This plugin needs access to user’s OIDC access token (this info is displayed to the user). As shown on the picture below, the RCauth plugin initially tries to get a proxy certificate from the credentials stored on the Credential Store (MyProxy server), where the user’s credentials are stored under unique user’s identifier. In case the credentials are not present in the Credential Store, the user is informed that it will be redirected to RCauth CA WAYF, where the user should use it’s home IdP to authenticate itself. For the test CA used in this demo, IdPs with low LoA can be used. After the user is successfully authenticated at the second IdP, the plugin uses new user's information to retrieve the certificate from the RCauth CA, stores it in the Credential Store (PUT or STORE flow in the picture), and creates a proxy certificate which is then, subsequently, provided to the user (GET flow). Proxy certificate can be retrieved as long as the original certificate is valid (~ 11 days), and itself is valid for 12 hours. Long lived proxy certificate obtained from the RCauth CA, along with the user’s private key, never leaves the Credential Store. Only Short Lived Proxy is provided to the user. Plugin also provides the possibility for the user to remove it’s credentials from the Credential Store. To demonstrate CLI capability for the user to retrieve a short proxy certificate, small bash script is provided.

The pilot structure is shown in picture below.
The pilot authentication flow is following:

1. User accesses (or is redirected to) the WaTTS page, and selects its OIDC provider to log in. This may be EGI, EUDAT, Google, Human Brain Project, INDIGO IAM or others.
2. User is redirected to the desired OIDC provider at which it authenticates. NOTE, the LoA is not relevant at this point (yet, since the demo uses the test CA).
3. User is prompted to accept the release of information, and at the end, the information about the user is returned to WaTTS (i.e. LoA, issuer, name, mail, etc.)
4. Back at WaTTS, user selects the X509_RCauth_Pilot
5. If already authenticated with RCauth see point 7. Otherwise, user is notified that he will be redirected to authenticate at RCauth. NOTE, this authentication step is only required once every 11 days, since that is the lifetime of the End Entity certificate.
6. User is redirected to RCauthWAYF and follows the respective "login-dance". NOTE: RCauth only accepts IdPs with an appropriate LoA.
7. User is presented with a page that allows him to download its X.509 proxy-certificate.

For commandline access with wattson this flow is applied:

4. User is also presented with his OIDC Access Token.
5a. If already authenticated, user may use wattson (or the nice cli-get-proxy.sh) to follow the commandline flow. User's proxy will be placed to /tmp /x509up_uxxxx.
5b. If not authenticated, user is informed that it has to authenticate by pointing its browser to WaTTS.

For obtaining a proxy via command line using SSH, the user needs to upload its SSH public key first. Afterwards, after first obtaining the proxy, the user can use its key to obtain a proxy. Separate plugin is provided for this access.

NOTE:
Names of the plugins may change.

WATTS source and info pages:

https://github.com/indigo-dc/tts
https://github.com/indigo-dc/wattson

Installing wattson: https://github.com/indigo-dc/wattson/releases/latest
https://github.com/urost/cli-get-proxy

Note that the WaTTS page might be moved in the future to https://github.com/watts-kit/watts.
Also, plugins can also be found here https://github.com/watts-kit/watts.

## Pilot resources

<table>
<thead>
<tr>
<th>WATTs service</th>
<th>(<a href="https://watts-dev.data.kit.edu">https://watts-dev.data.kit.edu</a>)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiple OIDC providers</td>
<td>(EGI CheckIn, EUDAT, Human Brain Project, IAM, Google)</td>
</tr>
<tr>
<td>myproxy server VM, running at KIT.</td>
<td></td>
</tr>
</tbody>
</table>

## Demonstration workflow

### Access to the cloud resources

1. **Access WaTTS instance at** [https://watts-dev.data.kit.edu](https://watts-dev.data.kit.edu)
   
   In this example we will use **EUDAT b2access**. So, we will select **“EUDAT (b2access)”** and click on **“Login”**.

2. **Select your Identity Provider** from the discovery page on the EUDAT b2access page.
   
   The institutional IdP to select (for this demo) is **KIT**, and then we click on **Authenticate**.
3. After authenticating, we can see what information is released to B2ACCESS.

4. After successful authentication, user is redirected from B2ACCESS to WaTTS. Then, in order to obtain a certificate user needs to use X509_RCauth_Pilot plugin, so we click on Request. The arrow is pointing to the information whether the credentials already exists on the myproxy-server or not. In this case, we will show the full flow, with authenticating with RCauth Test CA.

5. Since in this flow, the credentials are not yet stored, we will be redirected to authenticate with RCauth Test CA and obtain (new) credentials.
6. At this step, we are redirected to select the IdP with which to authenticate. If selecting the same IdP as used with the initial authentication at WaTTS (in this case home IdP), user is not prompted to authenticate again. However, at this point user can select different IdP to authenticate with RCauth. This flexibility provides the ability for the user to, e.g., use IdP with different LoAs, one to authenticate with WaTTS, different one to authenticate with RCauth. E.g. user can use Google account to authenticate with WaTTS, and use its home IdP to authenticate with RCauth.

7. Here, user can see what information will be released to the RCauth Test CA (and, in production to RCauth itself). User should click Yes if agrees with releasing the information.
8. User can now download and use the obtained short proxy certificate. The proxy certificate is also displayed to the user, (blacked in the example picture).