UVOS WEB REGISTRATION EXTENSION MANUAL

UNICORE Team

Document Version:	1.5.1
Component Version:	1.6.0
Date:	21 01 2013

This work is co-funded by the EC EMI project under the FP7 Collaborative Projects Grant Agreement Nr. INFSO-RI-261611.
This work was co-funded by the EC Chemomentum project under the FP6 Grant Agreement Nr. IST-033437.

Contents

UNICORE VO Service (UVOS) is a client-server system, developed to be used as an additional tool for large distributed systems, providing a solution for grid users management. Grid systems, especially UNICORE grid middleware, are the mainspring of the UVOS system. UVOS can be used with different systems, however is designed primarily to support UNICORE grid middleware.

For more information about UVOS visit http://uvos.chemomentum.org.

1 Introduction

VO authentication web component provides a login user WWW interface for the UVOS system. It used to provide SAML HTTP POST or HTTP Redirect binding implementation: other web components may redirect authentication requests to this component which authenticates the presenter and returns a well-formated, signed document certifying this fact.

The main features are:

- support for interactive email+password authentication
- support for authentication via web browser supplied certificate (i.e. client-authenticated TLS/SSL). In this case user is authenticated as UVOS DN identity or UVOS X.509 identity (anyone which will succeed).
- test mode: you can invoke the login servlet simply with HTTP GET. The authentication will be performed but no SAML response will be generated.

2 Compatibility

This release is compatible with UVOS 1.4.0 and 1.4.1 (future versions might be supported too).

3 Installation

UVOS web authentication extension is distributed either as a platform independent archive or as an installable, platform dependent package such as RPM. Depending on the installation source used, installation method and paths after installation are different.

3.1 Installation from RPM package (RedHat distributions)

The preferred way is to use Yum to install (and subsequently update) UVOS webauth.

To perform the Yum installation, EMI Yum repository must be installed first. Refer to the EMI release documentation (available at the EMI website http://www.eu-emi.eu/releases) for

detailed instructions. Typically installation of the EMI repository requires to download a single RPM file and install it.

After the EMI repository is configured, the following command installs UVOS webauth:

```
$> yum install unicore-uvos-webauth
```

The installed extension will be placed in the directory `/var/lib/unicore/uvos-server/-webapps/`

You have to restart UVOS server to complete the installation.

3.2 Installation from the DEB package (Debian distributions)

The preferred installation way is to use apt to install and subsequently update UVOS webauth.

To perform the apt installation, EMI apt repository must be installed first. Refer to the EMI release documentation (available at the EMI website http://www.eu-emi.eu/releases) for detailed instructions. Typically installation of the EMI repository requires to download a single DEB file and install it.

After the EMI repository is configured, the following command installs UVOS webauth:

```
$> apt-get install unicore-uvos-webauth
```

The installed extension will be placed in the directory `/var/lib/unicore/uvos-server/-webapps/`

You have to restart UVOS server to complete the installation.

3.3 Installation from the archive

If installing using a portable archive you have:

1. Download the archive from the UNICORE download site and unpack it.
2. In the unpacked archive you can find a file with a .war extension. Copy it to the webapps/ directory of UVOS server. This directory is placed directly under installation root folder which was used when installing UVOS server from portable archive.

You have to restart UVOS server to complete the installation.

4 Web browser path

The name of the war file is important as it is reflected in the URL by which web application is available. By default it is uvos-webauth-VERSION.war, where VERSION is e.g. "1.2". You can freely change the name of the war file. From now we will use label WAR_FILE_NAME to refer to actual base name of your war file.

You should be able to reach the web UI indirectly (via redirect) at the address:

```
<UVOS server base address>/WAR_FILE_NAME
```

or directly at:

```
<UVOS server base address>/WAR_FILE_NAME/VOauthentication.do
```

Example for the default server configuration and war named *uvos-webauth-1.0.war*:

```
https://localhost:2443/uvos-webauth-1.0
```

5 Configuration

It is possible to change the default HTML page used for the logging the user. To do so:

- create a new directory in UVOS server's `webapps` folder with desired name and unpack war archive into it.
- remove the unpacked war file from the `webapps` directory.
- modify CSS styles in template files located in `WEB-INF/templates`. Most of them are located in `headerTop.ftl` file.
- restart the UVOS server.

6 Developer howto

For the high level understanding how the authentication works see documents:

- SAML 2.0 core specification by OASIS - sec. 3.4 Authentication Request Protocol
- SAML 2.0 bindings specification by OASIS - sec. 3.4 HTTP Redirect Binding
- SAML 2.0 bindings specification by OASIS - sec. 3.5 HTTP POST Binding

To implement authentication in your servlet you have to:

- Create and return a XHTML document to the user (which should be authenticated). The document must contain POSTable form, with action equal to address of the web-auth servlet (contained in this package) installed at UVOS server. Check the Web Path Section 4 section to see what URL you should use. Most often JavaScript is used to submit the form automatically to the web-auth servlet.
- (and (hopefully) after while) Serve POST response with SAML response in it.

You can match request and response by so called *RelayState* data which can be assigned to the initial form and should be returned in the 2nd form.

UVOS client library offers a class that implements all the logic to perform the above pattern. The class is `pl.edu.icm.unicore.uvos.wsclient.samlapi.SAMLVOAuthnClientPOST` See its javadocs. Also there is a mostly complete servlet example in this package however note that it won't compile (that's why it isn't in sources) as it depends on uvos-client package which is not available (and needed) for the uvos-webauthn package. This example is located in the documentation directory and is called `TestLoginServlet.java`.