Single sign on: towards a new standard

able to allow apps and services to access easily private data

S. Bertocco

Funded by the European Union's
Horizon 2020 - Grant N° 824064

Current SSO recommendation

IVOA SSO describes how an IVOA application should apply a set of already standardized mechanisms to support single sign-on capabilities.
Approved standards for use in the SSO profile:

- No authentication required.
- HTTP Basic Authentication. *(RFC7235 updating RFC2617)*
- Transport Layer Security (TLS) with passwords. *(RFC5246)*
- Transport Layer Security (TLS) with client certificates. *(RFC5246 & RFC6818)*
- Cookies. *(RFC6265)*
- Open Authentication (OAuth). *(RFC6749)*
- Security Assertion Markup Language (SAML). *(saml-core-2.0-os OASIS standard)*
- OpenID. *(OpenID Foundation standards)*

IVOA service providers exposing secured services register in the IVOA registry metadata expressing conformance to one or more of the authentication mechanisms approved in the IVOA SSO profile using the securityMethod element.

Funded by the European Union’s Horizon 2020 - Grant N° 824064

Last discussion

Groningen Interop, Markus Demleitner Talk

Gaia, CADC, LSST use http request headers:

- Gaia and CADC Cookie with some custom name
- LSST RFC6750 authorization

Assuming all agree to use RFC 6750 (“Authorization: Bearer”),

Where to get the token?

Funded by the European Union’s
Horizon 2020 - Grant N° 824064

Last discussion: proposal

Introduce information on where to get the token with a tokenGetter element

```
<!-- Gaia: -->
<securityMethod standardID="ivo://ivoa.net/sso#RFC6750">
<tokenGetter type="userpass">https://<gaia>/tap/login
</tokenGetter></securityMethod>
<!-- CADC: -->
<securityMethod standardID="ivo://ivoa.net/sso#RFC6750">
<tokenGetter type="userpass">https://<cadc>/anywhere/login
</tokenGetter></securityMethod>
<!-- LSST: -->
<securityMethod standardID="ivo://ivoa.net/sso#RFC6750">
<tokenGetter type="manual">https://cilogin.org
</tokenGetter></securityMethod>
```

Funded by the European Union's
Horizon 2020 - Grant N° 824064

Open questions

- How to return token?
  in the html header or in the payload?

- securityMethod:tokenGetter should be 1:1 or 1:n ?
  Should we have an @title so clients can leave the choice to the user?

- How to manage federated authentication?
  Probably requiring the token to be a piece of signed (encoded) json

- How to manage credential delegation?

One more question:

- How to manage token refresh operation

Funded by the European Union's
Horizon 2020 - Grant N° 824064

Proposals

- How to return token?
  in the html header or in the payload?

  Patrick Dowler present now a proposal

Funded by the European Union's
Horizon 2020 - Grant N° 824064

Proposals

- securityMethod:tokenGetter should be 1:1 or 1:n ?
  Should we have an @title so clients can leave the choice to the user?

  I leave this point completely open to discussion

Funded by the European Union's
Horizon 2020 - Grant N° 824064

Proposals

- How to manage federated authentication?

  Probably requiring the token to be a piece of signed (encoded) json

  Proposal: use JSON Web Token (JWT) https://tools.ietf.org/html/rfc7519

  JSON Web Tokens are URL-safe JSON-based security tokens

  that contain a set of claims

  that can be signed and/or encrypted

  Example:

  a server could generate a token that has the claim "logged in as admin" and provide that to a client

Funded by the European Union's
Horizon 2020 - Grant N° 824064

Proposals

- How to manage credential delegation?

  Just a proposal:

  OAuth 2.0 Token Exchange
  https://tools.ietf.org/html/rfc8693

  OAuth 2.0 for Native Apps (Already cited in Groningen)
  https://tools.ietf.org/html/rfc8252

  The OAuth Security Model for Delegated Authorization
  https://tools.ietf.org/id/draft-barnes-oauth-model-01.html
  This document describes the security model for the OAuth authorization system, which allows a party that holds some authorization to delegate a subset of that authorization to another party, without requiring either party to disclose its credentials to the other.

Funded by the European Union's Horizon 2020 - Grant N° 824064

Proposals

- How to manage token renewal operation

  The OAuth 2.0 Authorization Framework
  https://tools.ietf.org/html/rfc6749#page-10

  Refresh Token

  Refresh tokens are credentials used to obtain access tokens. Refresh tokens are issued to the client by the authorization server and are used to obtain a new access token when the current access token becomes invalid or expires

Funded by the European Union's Horizon 2020 - Grant N° 824064