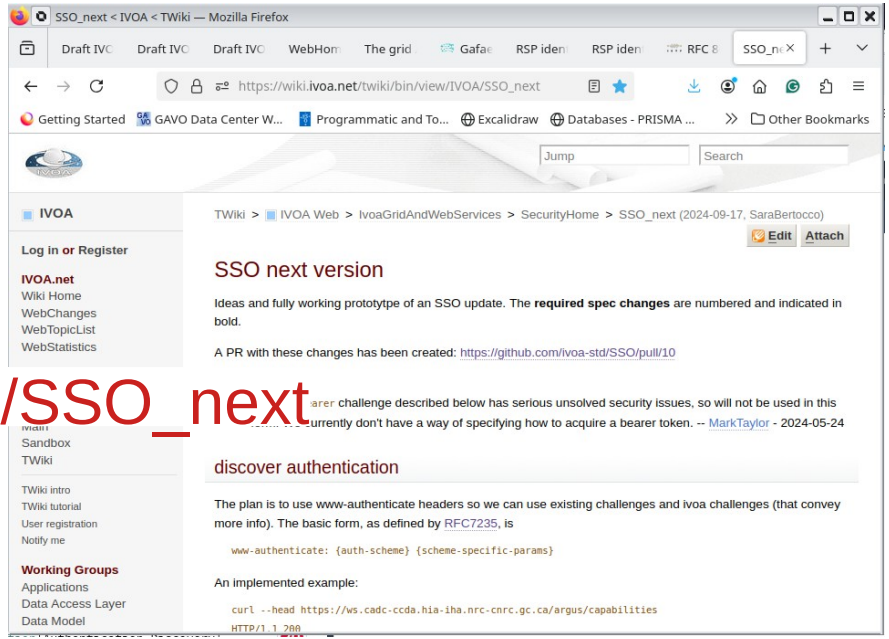
SSO status & SSO-next

Ongoing discussion: e-mail threads

E-mail threads subjects (http://mail.ivoa.net/pipermail/grid/)

- “Authentication and DataLink” (2020)
- “Authentication progress” (2020)
- “authcheck endpoint” (2021)
- “SSO next” (2022) → https://wiki.ivoa.net/twiki/bin/view/IVOA/SSO_next
- https://github.com/ivoa-std/SSO/issues
- “SSO major editing” (2024)
- “ivoa-oauth: an SSO-next based approach to allowing non-browser-based VO clients to use OAuth 2.x/OIDC” (2024)



Ongoing discussion: presentations

- Patrick Dowler strawman proposal:

https://wiki.ivoa.net/internal/IVOA/InterOpMay2020GWS/auth-reqs-strawman.pdf

- Mark Taylor:

https://wiki.ivoa.net/internal/IVOA/InterOpNov2020GWS/auth.pdf

https://wiki.ivoa.net/internal/IVOA/InterOpNov2023Apps/auth.pdf

Strawman for using HTTP headers

- define {challenge} for each credential type -- use SSO identifiers or industry standard strings where available, e.g.

```
WWW-Authenticate: "bearer" realm="foo",
    "ivo://ivoa.net/std/SSO#cookie", "ivo://ivoa.net/std/SSO#tls-with-certificate"
```

- use challenge params to convey the required info, e.g.

```
WWW-Authenticate: ivo://ivoa.net/std/SSO#cookie accessURL="https://example.net/login"
    securityMethod="ivo://ivoa.net/std/SSO#BasicAA" [, {challenge}, . . .]
```

- feasible to co-exist with and/or describe industry standards... not sure about adding params to existing challenges though
- prior art: https://tools.ietf.org/id/draft-broyer-http-cookie-auth-00.html

Challenge syntax

Challenge syntax

- General format:

  `WWW-Authenticate: <auth-scheme-name> <scheme-params>`

- `<auth-scheme-name>` must match RFC7235 *token* syntax
  - "`ivo://ivoa.net/std/SSO#cookie`" not syntactically legal (contains "/")
  - Could use:
    - `WWW-Authenticate: vo-sso-cookie`
    - `WWW-Authenticate: vo-sso securityMethod="ivo://ivoa.net/std/SSO#cookie"`
    - ... or something else
- Easy to solve — just choose one
- There is an IANA registry of these `<auth-scheme-names>`, but we probably don't need to register them

Consensus substantial enough

- SecurityMethod definitions no longer required
- The autentication is conveyed using WWW-Authenticate challenges as defined by IETF RFC 7235
- Clients bootstrap making anonymous HTTP GET or HTTP HEAD requests to the service's /capabilities endpoint.
- Response HTTP status codes 200 or 401/403 with

  or without a supported challenge
- Basic form of the header in an authentication challenge is:

  www-authenticate: auth-scheme scheme-specific-params

Bootstrap

curl sI http://dc.g-vo.org/tap/capabilities

```
HTTP/1.1 200 OK
Server: DaCHS/2.10 twistedWeb/22.4.0
Date: Thu, 07 Nov 2024 15:27:07 GMT
WWW-Authenticate: Basic realm="Gavo"
Content-Type: text/xml
```

HTTP/1.1

200 with supported challenge, default No Auth, but Optional Auth

200 with no (supported) challenge, No Auth

401/403 with supported challenge, Mandatory Auth

Challenges

- WWW-Authenticate: ivoa_cookie

  standard_ id="ivo://ivoa.net/sso#tls-with-password",

  access_url="https://geadev.esac.esa.int/tap-server/login"

- WWW-Authenticate: ivoa_x509

  standard_id="ivo://ivoa.net/sso#BasicAA",

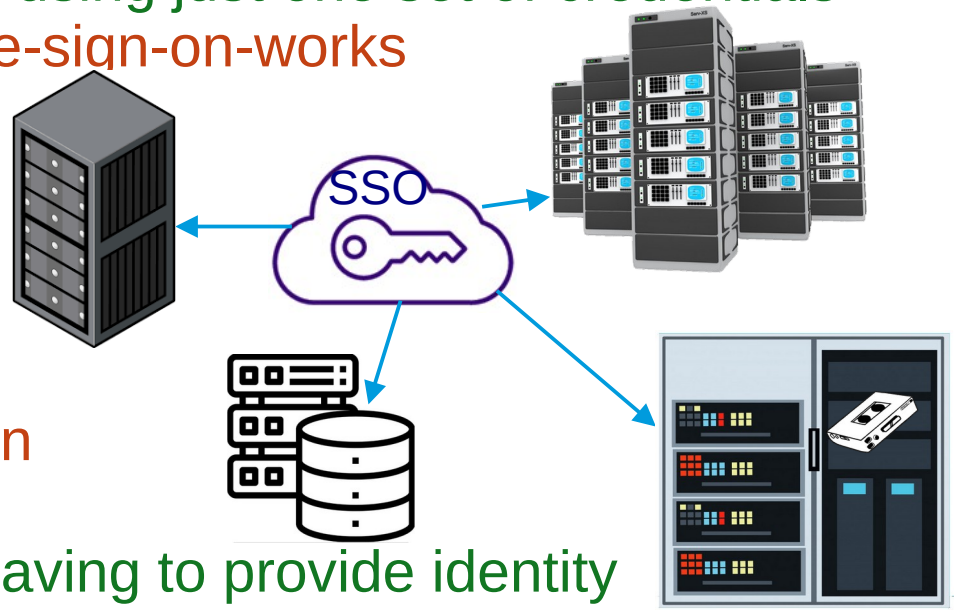
  access_url="https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/cred/auth/priv"

- WWW-Authenticate: Basic realm="Gavo"
- IVOA_oauth ← To be discussed

Is SSO-next a Single Sign-On ?

SSO is an *authentication* method that enables users to securely *authenticate* with multiple applications and websites by using just one set of credentials
https://www.onelogin.com/learn/how-single-sign-on-works

Single sign-on (SSO) is an authentication scheme that allows a user to log in with a single ID to any of several related, yet independent, software systems
https://en.wikipedia.org/wiki/Single_sign-on

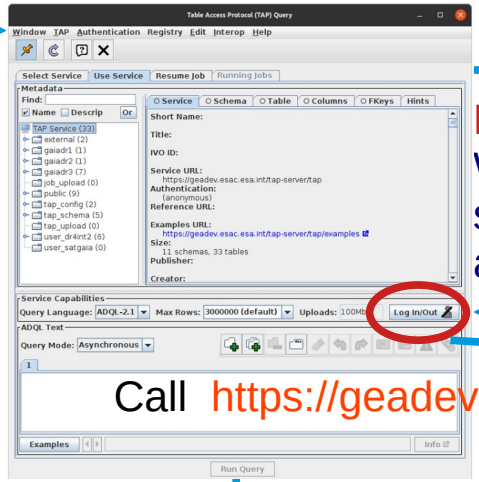


From a user's perspective it means only having to provide identity credentials once to access a number of different services within a particular realm - the greater the reach of the realm the more useful the "SSO" [Paul Harrison]

Example of challenge

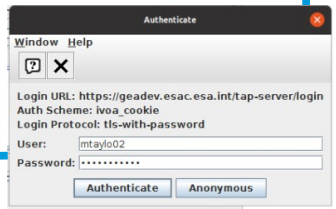
Search data

Topcat



Call capabilities
https://geadev.esac.esa.int/tap-server/tap/capabilities

HTTP/1.1 401
WWW-Authenticate: ivoa_cookie
standard_id="ivo://ivoa.net/sso#tls-with-password",
access_url="https://geadev.esac.esa.int/tap-server/login"



Return authentication cookie

Authenticated user access data

Archive

```
<?xml version="1.0"?>
<VOTABLE version="1.3" xmlns="http://www.ivoa.net/xml/VOTable/v1.3"
xmlns:dl="http://www.ivoa.net/xml/Datalink/v1.0">
  <RESOURCE type="results">
    <DESCRIPTION>Example of VOTable with DataLink</DESCRIPTION>
    <TABLE>
      <FIELD name="Name" datatype="char" arraysize="*" />
      <FIELD name="DataLink" datatype="char" ucd="meta.ref.url" arraysize="*" />
      <DATA>
        <TABLEDATA>
          <TR>
            <TD>Sirius</TD>
            <TD>https://example.org/data/sirius</TD>
          </TR>
          <TR>
            <TD>Betelgeuse</TD>
            <TD>https://archive.org/data/betelgeuse</TD>
          </TR>
```

Call https://example.org/data/capabilities
WWW-Authenticate: Basic realm="example"
WWW-Authenticate: access_url=”example.login”
Call https://archive.org/data/capabilities
WWW-Authenticate: Basic realm="archive"
WWW-Authenticate: access_url="archive.login”

A new title for a new specification?

“So I make the provocative suggestion that the SSO document should be rewritten more or less from scratch, ..., describing only(?) how non-browser clients can interact with authenticated services in a VO-standard way. It should describe in detail how such interaction works, but only for those authentication methods where we actually understand how to do it (currently: BasicAA, cookies and X.509 certificates; hopefully bearer tokens will be added at some point) ……..The document would perhaps acquire a new name or become a different document in the process.” [Mark Taylor]

A new title for a new specification?

“IVOA cookie+client cert, plus specifying how to do discovery” [James Tocknell]

“perhaps the whole aim should be an SSO “endorsed note” as we should only be describing patterns of using existing protocols rather than defining our own.” [Paul Harrison]

Proposal

- A new document

  title: “IVOA Interoperable Authentication Management”

  content:
  - The authentication process
    - Bootstrapping
    - Authentication discovery
    - Challenge syntax
  - Supported challenges:
    - Basic
    - ivoa-cookies
    - ivoa-x509
    - ivoa-oauth (?)

Feedback Discussion

Backup slides

Pat Dowler

Strawman for using HTTP headers

- define {challenge} for each credential type -- use SSO identifiers or industry standard strings where available, e.g.

```
WWW-Authenticate: "bearer" realm="foo",
     "ivo://ivoa.net/std/SSO#cookie", "ivo://ivoa.net/std/SSO#tls-with-certificate"
```

- use challenge params to convey the required info, e.g.

```
WWW-Authenticate: ivo://ivoa.net/std/SSO#cookie accessURL="https://example.net/login"
      securityMethod="ivo://ivoa.net/std/SSO#BasicAA" [, {challenge}, . . .]
```

- feasible to co-exist with and/or describe industry standards... not sure about adding params to existing challenges though
- prior art: https://tools.ietf.org/id/draft-broyer-http-cookie-auth-00.html

Mark Taylor

Challenge syntax

Challenge syntax

- General format:

  `WWW-Authenticate: <auth-scheme-name> <scheme-params>`

- `<auth-scheme-name>` must match RFC7235 *token* syntax
  - "`ivo://ivoa.net/std/SSO#cookie`" not syntactically legal (contains "/")
  - Could use:
    - `WWW-Authenticate: vo-sso-cookie`
    - `WWW-Authenticate: vo-sso securityMethod="ivo://ivoa.net/std/SSO#cookie"`
    - ... or something else

- Easy to solve — just choose one
- There is an IANA registry of these `<auth-scheme-names>`, but we probably don't need to register them

Mark Taylor

WWW-Authenticate Challenges

The following RFC 7235 challenge authentication schemes are currently recognised:

`basic`

- This is simply HTTP Basic authentication RFC 7617, no VO customisation required
- Supported by DaCHS (and others?)

```
% curl -sI http://dc.g-vo.org/tap/capabilities | grep Basic
WWW-Authenticate:  Basic realm="Gavo"
```

`ivoa_cookie`

- Uses RFC 2965 cookies, with additional parameters indicating where/how to get a cookie
- Supported by ESDC absi-lib-tap library v9.4.0
  - Mostly deployed internally at ESAC
  - Public TAP deployments: Gaia dev service (geadev) now, Euclid & PDS soon?, Gaia public service eventually?

```
% curl -sI https://geadev.esac.esa.int/tap-server/tap/capabilities | grep ivoa_cookie
WWW-Authenticate:  ivoa_cookie standard_id="ivo://ivoa.net/sso#tls-with-password", access_url="https://geadev.esac.esa.int/tap-server/login"
```

`ivoa_x509`

- Uses X.509 certificates, with additional parameters indicating where/how to get one
- Supported by CADC services

```
% curl -sI https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/argus/capabilities | grep ivoa_x509
www-authenticate:  ivoa_x509 standard_id="ivo://ivoa.net/sso#BasicAA", access_url="https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/cred/auth/priv"
www-authenticate:  ivoa_x509
```

~~`ivoa_bearer`~~

- Some way to work with OAuth2.0/RFC 6750 Bearer Tokens would be good...
- ... but currently no way to acquire tokens securely, so **not supported**
- Hopefully some progress in future (RFC 8252? token scope labelling?), driven by service implementations

- those who have OAuth/OIDC servers should document what they're using so we can see what the common baseline is?
- Should we make a page under the SSO next page where we tabulate that?
- ---------
- This has also been my personal feeling about an SSO “standard” from the early days - SSO is
- in general driven by “market forces” outside the IVOA control - so that perhaps the whole aim should be
- an SSO “endorsed note” as we should only be describing patterns of using existing protocols rather than defining our own. Paul Harrison
- I don't see much
- point in providing an "approved" list of web-based technologies
- for authentication. Mark Taylor
- What we do need is the description of how to initiate authenticated
- access to VO services from non-browser clients, since there are in
- general not standards on the wider internet about how to do this.