Authentication for Non-Browser Clients: Update

Mark Taylor (Bristol)

with input from: Brian Major (CADC)

Pat Dowler (CADC)

Markus Demleitner (ARI)

GWS WG

IVOA Interop

Online

4 November 2021

`$Id: auth.tex,v 1.13 2021/11/03 18:28:14 mbt Exp $`

History

Working towards authenticated access for non-browser clients

- Clients need to know **how** and **where** (and **whether**) to log in for authenticated access

Previous updates:

- May 2020
  - ▷ Pat Dowler: *Authentication strawman proposal*
- Nov 2020
  - ▷ Brian Major: *Non-browser client authentication with OAuth2 tokens*
  - ▷ Mark Taylor: *Authentication Implementation Report*
- May 2021
  - ▷ Brian Major: *Tokens for Non-Browser Clients Updates*

Recent activity:

- Implementation in CADC services
- Implementation in TOPCAT client
- Becoming clearer what will work?

Current Status

How to communicate authentication methods to client?

- Challenge-based auth method declaration seems like the way to go (RFC7235 — `WWW-Authenticate` headers)
- `<securityMethod>` elements in VOSI-capabilities document not so useful?
  - ▷ would only duplicate information that has to be in challenge anyway

Some details still to decide on:

- Exact form of challenges
  - ▷ Bearer token type
  - ▷ Cookie type
  - ▷ Others?
- Bootstrap challenge location
  - ▷ On new dedicated (`/authcheck`?) endpoint?
  - ▷ On existing `/capabilities` endpoint?

... with adjustments?

More implementations required

- especially cookies

Standardisation required

- SSO update?

Example Bearer Token Mechanism

Currently implememented at CADC:

```
% curl -I https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/argus/capabilities
HTTP/1.1 200
www-authenticate: ivoa standard_id="ivo://ivoa.net/sso#tls-with-password", access_url="https://ws-cadc.canfar.net/ac/login"
www-authenticate: ivoa standard_id="ivo://ivoa.net/sso#OAuth", access_url="https://ws-cadc.canfar.net/ac/authorize"
www-authenticate: Bearer
www-authenticate: ivoa standard_id="ivo://ivoa.net/sso#tls-with-certificate"
content-type: text/xml
transfer-encoding: chunked
date: Mon, 18 Oct 2021 12:21:56 GMT

% curl -D - -d username=mbt -d password=xxxx https://ws-cadc.canfar.net/ac/login
HTTP/1.1 200
x-vo-authenticated: mbt
content-type: text/plain;charset=ISO-8859-1
content-length: 767
date: Mon, 18 Oct 2021 12:26:07 GMT

base64:ZXhwaXJ5dGltZT0xNjM0NzMyNzY4MTE0Jm51bWVyaWNJRD0wMDAwMDAwMC0wMDAwLTAwMDAtMDAwMC0wMDAwMDc1NDA4ZTQmUE9TSVg9MTIyOTQ3ODEyJlg1MDA9Q049bWJ0XzB...

% curl -D - --header 'Authorization: ivoa base64:ZXhwaXJ5dGltZT0x...' https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/argus/async
HTTP/1.1 200
x-vo-authenticated: mbt
content-type: text/xml
transfer-encoding: chunked
date: Mon, 18 Oct 2021 12:44:57 GMT

<?xml version="1.0" encoding="UTF-8"?>
<uws:jobs xmlns:uws="http://www.ivoa.net/xml/UWS/v1.0" xmlns:xlink="http://www.w3.org/1999/xlink" version="1.1">
  <uws:jobref id="jfau8kfbsustskid">
    ...
```

Example Bearer Token Mechanism

Currently implememented at CADC:

```
% curl -I https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/argus/capabilities
HTTP/1.1 200
www-authenticate: ivoa standard_id="ivo://ivoa.net/sso#tls-with-password", access_url="https://ws-cadc.canfar.net/ac/login"
www-authenticate: ivoa standard_id="ivo://ivoa.net/sso#OAuth", access_url="https://ws-cadc.canfar.net/ac/authorize"
www-authenticate: Bearer
www-authenticate: ivoa standard_id="ivo://ivoa.net/sso#tls-with-certificate"
content-type: text/xml
transfer-encoding: chunked
date: Mon, 18 Oct 2021 12:21:56 GMT

% curl -D - -d username=mbt -d password=xxxx https://ws-cadc.canfar.net/ac/login
HTTP/1.1 200
x-vo-authenticated: mbt
content-type: text/plain;charset=ISO-8859-1
content-length: 767
date: Mon, 18 Oct 2021 12:26:07 GMT

base64:ZXhwaXJ5dGltZT0xNjM0NzMyNzY4MTE0Jm51bWVyaWNJRD0wMDAwMDAwMC0wMDAwLTAwMDAtMDAwMC0wMDAwMDc1NDA4ZTQmUE9TSVg9MTIyOTQ3ODEyJlg1MDA9Q049bWJ0XzB...

% curl -D - --header 'Authorization: ivoa base64:ZXhwaXJ5dGltZT0x...' https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/argus/async
HTTP/1.1 200
x-vo-authenticated: mbt
content-type: text/xml
transfer-encoding: chunked
date: Mon, 18 Oct 2021 12:44:57 GMT

<?xml version="1.0" encoding="UTF-8"?>
<uws:jobs xmlns:uws="http://www.ivoa.net/xml/UWS/v1.0" xmlns:xlink="http
  <uws:jobref id="jfau8kfbsustskid">
    ...
```

Some things to iron out:

- Scope of retrieved token (reuse for other URLs)
- Exact form of token delivery (body/header?)
- Exact form of `Authorization` header (“`ivoa`” / “`Bearer`”?)

Possible Cookie Mechanism

Not implemented yet, but suggest something like:

```
% curl -I https://example.com/tap/capabilities
HTTP/1.1 200
www-authenticate: ivoa standard_id="ivo://ivoa.net/sso#cookie", access_url="https://example.com/cookie-login"
content-type: text/xml

% curl -I -d username=mbt -d password=xxxx https://example.com/cookie-login
HTTP/1.1 200
x-vo-authenticated: mbt
set-cookie: JSESSIONID=3B9C6A931B6871C87E2403A791F1E86D; Path=/tap; Secure; HttpOnly

% curl -D - --header "Cookie: JSESSIONID=3B9C6A931B6871C87E2403A791F1E86D" https://example.com/tap/async
HTTP/1.1 200
x-vo-authenticated: mbt
content-type: text/xml

<?xml version="1.0" encoding="UTF-8"?>
<uws:jobs xmlns:uws="http://www.ivoa.net/xml/UWS/v1.0" xmlns:xlink="http://www.w3.org/1999/xlink" version="1.1">
  <uws:jobref id="jfau8kfbsustskid">
    ...
```

Form of Challenges

Currently VO-specific challenges at CADC look like:

```
www-authenticate:  ivoa standard_id="ivo://ivoa.net/sso#tls-with-password", access_url="https://ws-cadc.canfar.net/ac/login"
www-authenticate:  ivoa standard_id="ivo://ivoa.net/sso#tls-with-certificate"
www-authenticate:  ivoa standard_id="ivo://ivoa.net/sso#cookie", access_url="https://...."
```

Maybe better?

```
www-authenticate:  ivoa-bearer access_url="https://ws-cadc.canfar.net/ac/login"
www-authenticate:  ivoa-certificate
www-authenticate:  ivoa-cookie access_url="https://...."
```

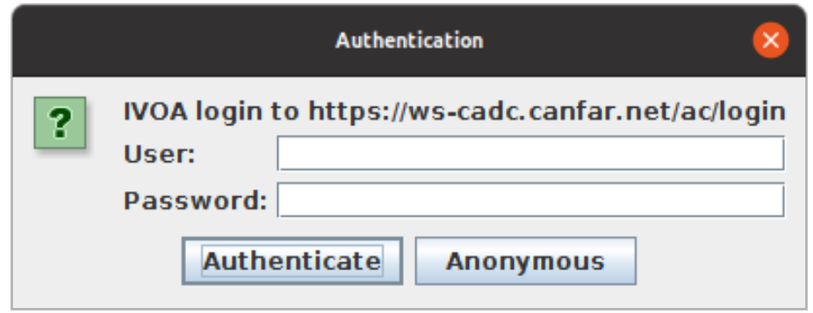
How much flexibility is required here?

- Client needs to know:
  - How to transmit credentials (e.g. API + URL)
  - How to extract authentication information from response (e.g. what header contains Bearer token)
- Should these be coupled or decoupled?
  - Do we need to mix'n'match credential-supply and authentication-retrieval information?

Bootstrap Challenge Location

Where do clients look for `WWW-Authenticate` challenges?

- Challenges will be present on all endpoints
- (Some) clients want to set up authentication before other service interaction
- Which endpoint to use?
  - ▷ `/capabilities`
    - Always present
    - **But** must be anonymously readable (TAP 1.1 sec 2)
      ⇒ cannot return `401 Unauthorized`
      ⇒ client/user can't tell whether authentication is optional or mandatory
  - ▷ `/authcheck` *(or something)*
    - New no-op endpoint that just provides challenges
    - Seems annoying to add a new endpoint only for auth
    - But does permit clients to distinguish optional/mandatory authentication
  - ▷ `/capabilities` with changes
    - Permit auth-only (`401`) access, require HTTP HEAD
    - Anonymous access no longer necessary, since `securityMethods` no longer used by clients
    - Would require some standards changes (VOSI, TAP)
    - Solves optional/mandatory distinction
  - ▷ Something else?
    - Can't think of a suitable existing endpoint/service URL for this



What happens if I hit anonymous?
Unprivileged access or can't use the service?

Next Steps

Agree on questions:

- Bootstrap challenge endpoint: `/capabilities`, `/authcheck`, other?
- General form of challenges: couple/decouple login API & token return?
- *Bearer Token* challenge/response details: CADC implementation OK? Scope?
- *Cookie* challenge/response details: draft and agree
- Other challenge/response types?
- Retire/ignore `<securityMethod>` elements from VOSI-capabilities?

Implementations:

- CADC already has tls-with-password/Bearer (on `/capabilities`)
- GAVO/DaCHS has Basic Auth (on `/authcheck`)
- Cookie challenge implementation required — CADC? ESDC?
- Others?
- TOPCAT/Auth library ready for client side implementation

Draft/write standards text

- Additional text + some reorganisation in SSO (v2.1? v3.0?)?
- New document?

(Not necessarily in that order)