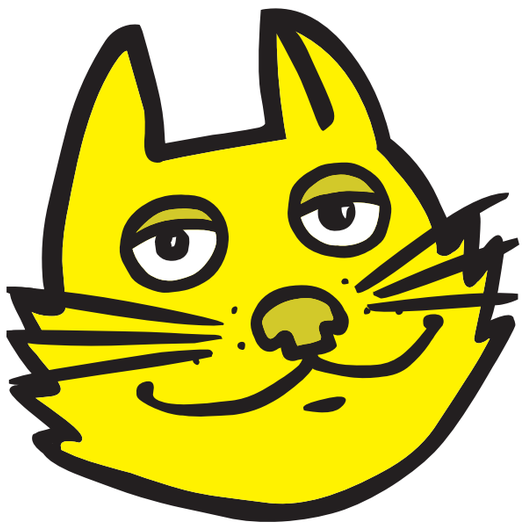
Authentication in TOPCAT



Mark Taylor (University of Bristol)

Applications WG
IVOA Interop
Tucson AZ

12 November 2023

$Id: auth.tex,v 1.8 2023/11/02 15:50:41 mbt Exp $

You can now access authenticated services (TAP/DataLink/other) from TOPCAT

You can now access authenticated services (TAP/DataLink/other) from TOPCAT

(as long as the service conforms to the SSO_next proposal)

You can now access authenticated services (TAP/DataLink/other) from TOPCAT

(as long as the service conforms to the SSO_next proposal)

(which currently exists on a wiki page and unpublished SSO WD)

SSO_next Proposals

Summary (SSO_next wiki page, main author Pat Dowler):

- Communicate auth info using only RFC 7235-style `WWW-Authenticate` challenges
  - `securityMethod` elements in capabilities document no longer required
- Introduce VO-specific `ivoa_*` authentication schemes that can carry additional metadata:
  - `access_url`: where to login
  - `standard_id`: how to interact with `access_url`
- Provide authentication confirmation
  - `X-VO-Authenticated` header SHOULD appear in authenticated responses
- Specify how to determine service authentication requirements
  - `/capabilities` endpoint `HEAD`/`GET` response is 200/401(/403) with or without `WWW-Authenticate` challenge(s)

Impact on standards:

1. VOSI augmented to require HTTP `HEAD` support for `/capabilities` endpoints
2. SSO augmented to define `ivoa_cookie`, `ivoa_x509` and ~~`ivoa_bearer`~~ challenges
3. SSO `ivo://ivoa.net/sso#tls-with-password` extended to specify the form params
4. SSO challenges describe/specify response form of login endpoint (`access_url`)
5. SSO requires/recommends authenticated services to include `x-vo-authenticated` header in all authenticated responses
6. Modify VOSI to allow `/capabilities` to respond with 401 (or 403) *(also affects TAP 1.1 sec 2; & others?)*

TOPCAT Auth Behaviour

- Reactive:
  - If it encounters a 401/403 response with a supported challenge, seeks username/password from user, and retries
- Pre-emptive (currently TAP only):
  - When starting to use service, call HTTP HEAD on `/capabilities`:
    - ▷ If 401/403 with supported challenge, seeks username/password from user (**Mandatory Auth**)
    - ▷ If 200 with supported challenge, default to unauthenticated use, but provide optional login button (**Optional Auth**)
    - ▷ If 200 with no (supported) challenge, continue unauthenticated (**No Auth**)
- Proactive:
  - Once authenticated to some domain, automatically supply credentials with subsequent HTTP requests to the same domain

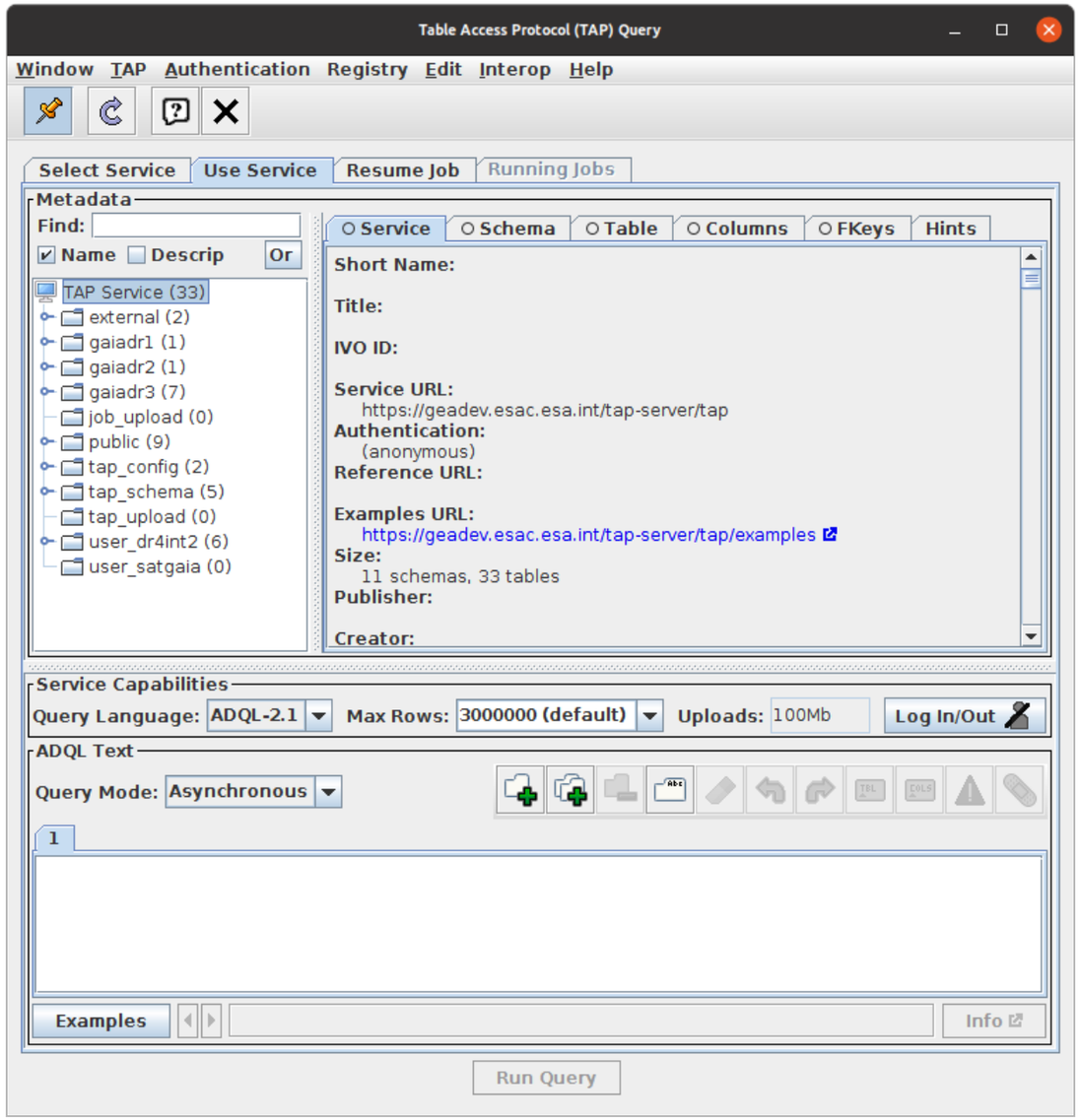
Glossary:

challenge
: is an RFC 7235 `WWW-Authenticate` HTTP response header indicating accepted authentication options

supported challenge
: is a challenge that the AUTH library understands (currently `ivoa_cookie`, `ivoa_x509`, `basic`)

domain
: is a set of resources with the same authentication arrangements; membership rules are challenge-type specific

TAP Pre-emptive Authentication



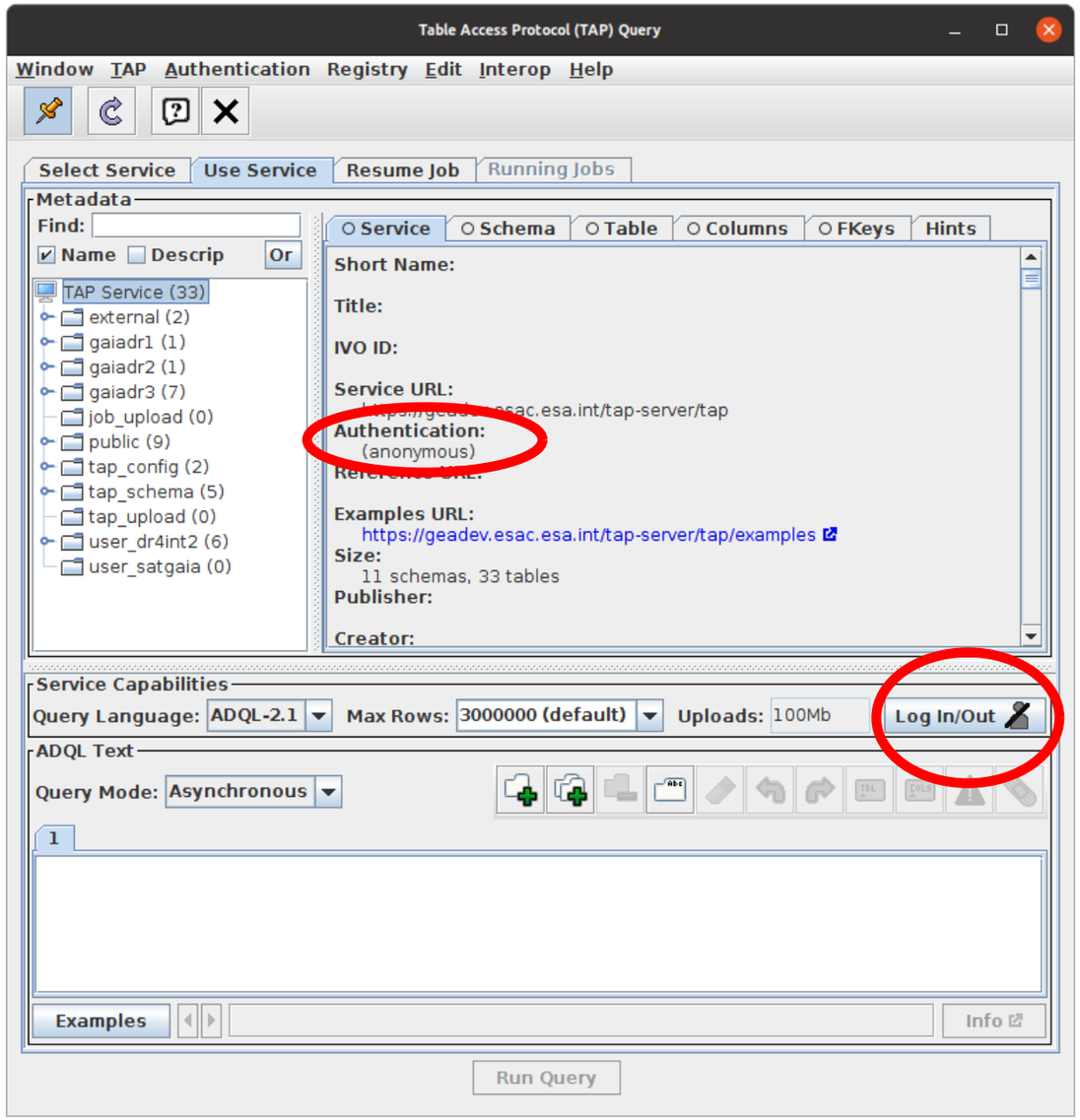
Effects of TAP auth are service-dependent. Maybe:

- You are allowed to use the service (mandatory auth)
- UWS documents (async TAP query+result pages) are protected
- Access to restricted data
- Elevated user resource limits

Once authenticated, SSO applies

- You can access other resources from the same domain (e.g. DataLink)

TAP Pre-emptive Authentication



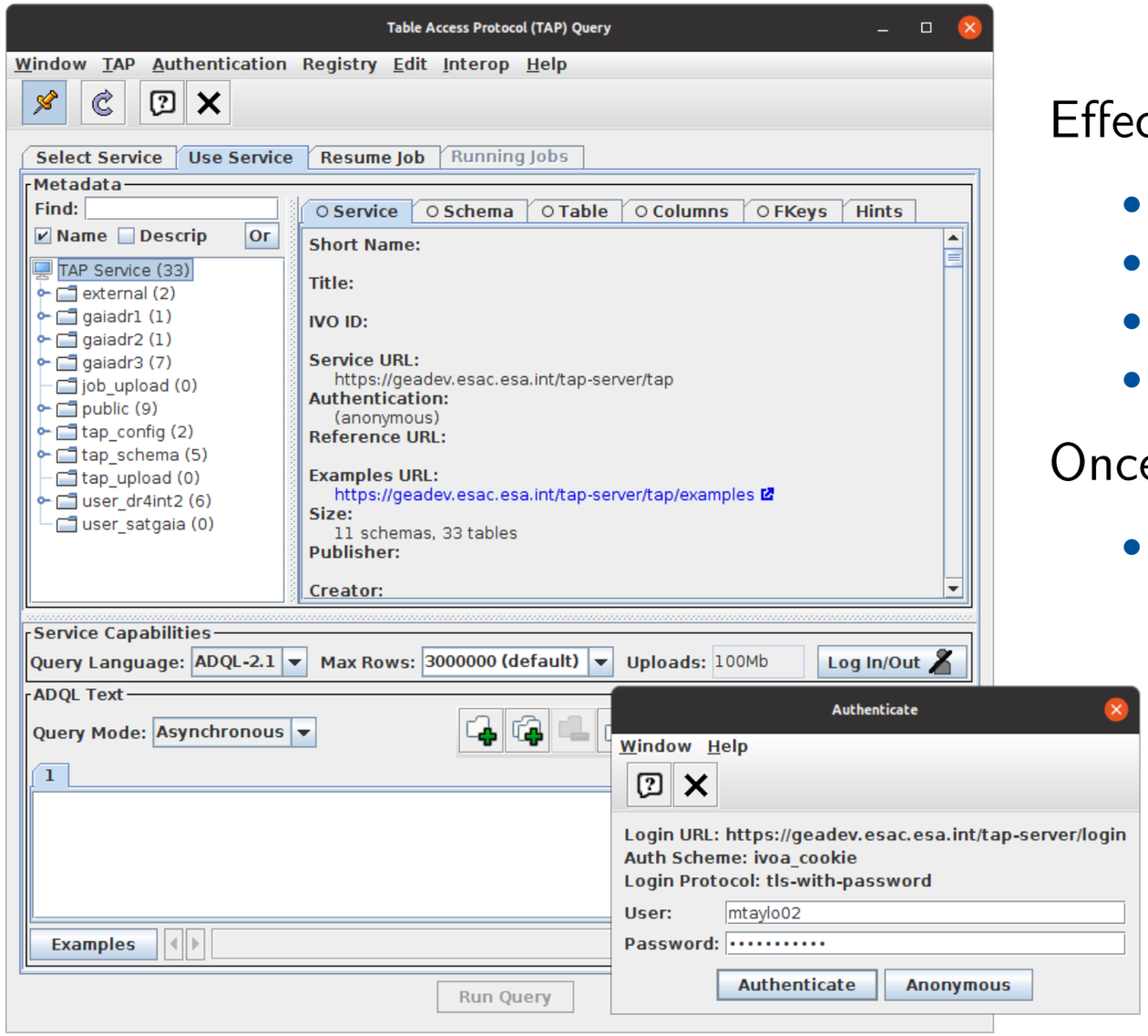
Effects of TAP auth are service-dependent. Maybe:

- You are allowed to use the service (mandatory auth)
- UWS documents (async TAP query+result pages) are protected
- Access to restricted data
- Elevated user resource limits

Once authenticated, SSO applies

- You can access other resources from the same domain (e.g. DataLink)

TAP Pre-emptive Authentication



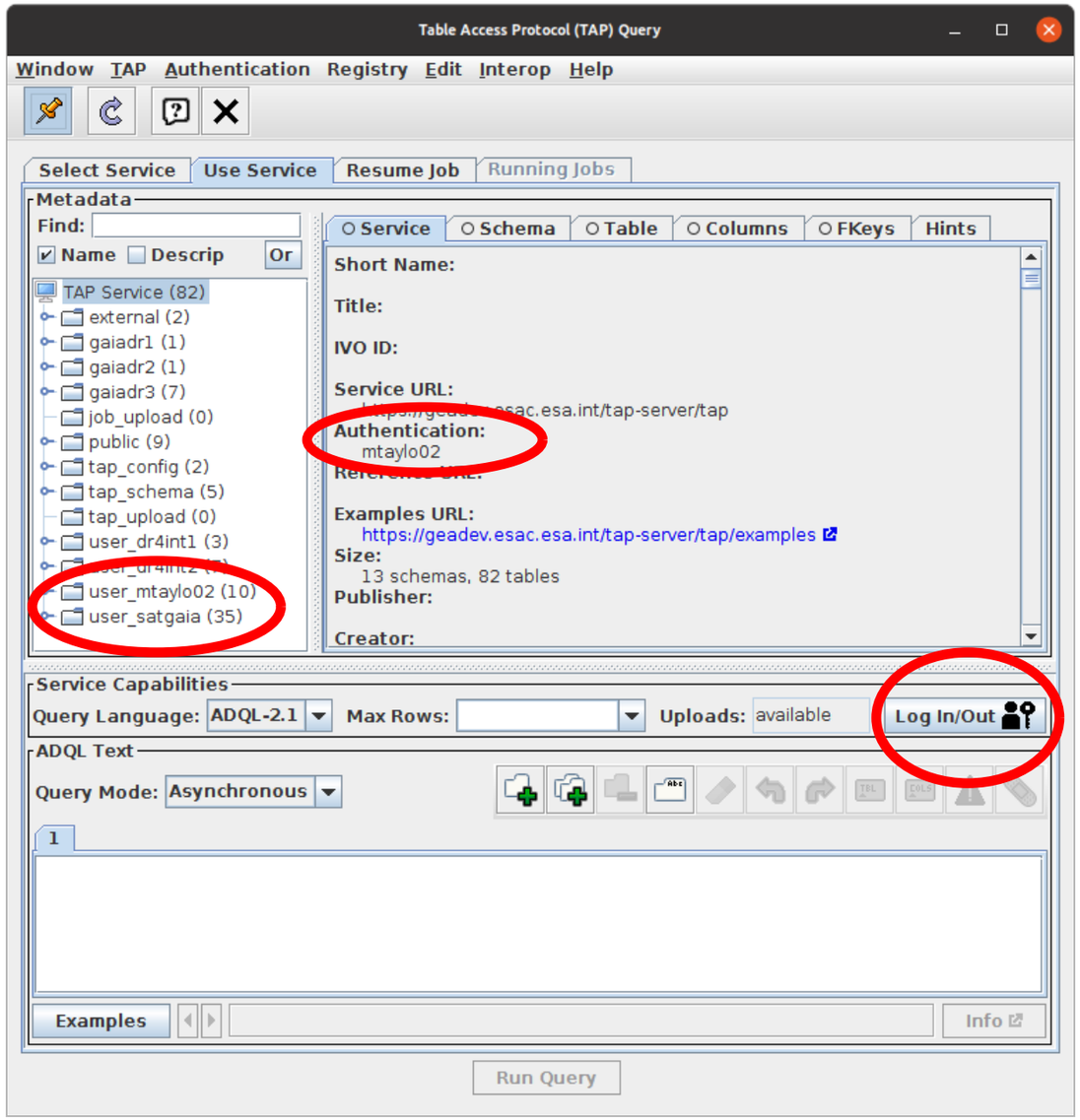
Effects of TAP auth are service-dependent. Maybe:

- You are allowed to use the service (mandatory auth)
- UWS documents (async TAP query+result pages) are protected
- Access to restricted data
- Elevated user resource limits

Once authenticated, SSO applies

- You can access other resources from the same domain (e.g. DataLink)

TAP Pre-emptive Authentication



Effects of TAP auth are service-dependent. Maybe:

- You are allowed to use the service (mandatory auth)
- UWS documents (async TAP query+result pages) are protected
- Access to restricted data
- Elevated user resource limits

Once authenticated, SSO applies

- You can access other resources from the same domain (e.g. DataLink)

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

AUTH Library Usage

TOPCAT uses a new internal library AUTH to manage HTTP interactions

Authentication is managed by an application-wide `AuthManager` instance

- Initialisation:

  ```
  AuthManager authManager = AuthManager.getInstance();
  authManager.setUserInterface(UserInterface.GUI); ( or CLI)
  ```

- Replace all URL accesses that may be authenticated with AuthManager equivalents
  - ▷ Use drop-in replacement for `java.net.URL.openStream()`:

    ```
    InputStream in = authManager.openStream(url);
    ```

  - ▷ Other methods are available for more complicated HTTP interactions (POST, redirects, header manipulations etc)
- Optionally provide pre-emptive log in to TAP-like services:

  ```
  AuthStatus status = authManager.authcheck(capabilitiesUrl, isHead, isForceLogin);
  ```

- AuthManager keeps track of per-domain auth contexts:
  - ▷ makes connections
  - ▷ watches for recognised WWW-Authenticate challenges
  - ▷ interacts with users/services to acquire credentials
  - ▷ uses cached credentials for subsequent requests to related URLs
  - ▷ (also handles 3xx redirects, POST connections etc which need to manage challenges/credentials; somewhat messy)

AUTH Library Deployment

AUTH java library for VO-compliant authentication

- Small standalone jar file (90kb)
- No external dependencies
- Currently available as part of STIL v4.2
  - ▷ Javadocs available on web page
  - ▷ File `auth.jar` can be extracted from stil_jars.zip
- May be published in future as a separate product (including Maven central repo) depending on demand
- If you're interested in using it, talk to me

Advice for Clients

How do I get my client to talk to authenticated VO services?

- Desktop clients (Java)
  - ▷ Consider using the AUTH library (talk to me)
- Desktop clients (other languages)
  - ▷ ??
- Web clients
  - ▷ Different approaches probably more appropriate
  - ▷ It seems like most modern auth is expecting to be used from the web

Advice for Services

How do I get TOPCAT to authenticate against my service?

- Implement one of the existing SSO_next auth schemes (`ivoa_cookie`, `ivoa_x509`, `basic`)
  - ▷ This may be as simple as adding a suitable `WWW-Authenticate` header to the `/capabilities` response
- Prototype some other auth scheme (OAuth2.0/Bearer Tokens?) and discuss within GWS
  - ▷ I plan to update the AUTH library as standards evolve

Next Steps

- Standardisation
  - Withdraw or redefine `ivoa_bearer` scheme in SSO (it's insecure as currently specified)
  - Provide a published SSO WD with details of SSO_next requirements
  - Figure out how to cope with OAuth2/Bearer Tokens
  - Introduce new `ivoa_*` auth schemes as required
- Implementation
  - Encourage service providers to comply with existing/developing recommendations
  - Other clients adopt AUTH library? or make their own arrangements
  - AUTH library published as standalone component if there is demand