Authentication Implementation Report

Mark Taylor (Bristol)

GWS WG
IVOA Virtual Interop

19 November 2020

$Id: auth.tex,v 1.9 2020/11/18 10:09:29 mbt Exp $

Outline

- (My) client requirements
- Context: SSO 2.0, TAP 1.1, WWW-Authenticate strawman
- Issues: challenge syntax, auth bootstrapping
- Implementation status: client library, service status
- Summary + outlook

Client Requirements

TOPCAT's eye view

- Two resource access scenarios:
  - ▷ User wants to use TAP (or TAP-like?) service
    1. Establish auth context
    2. Interact with service (acquire service metadata, run queries, etc)
  - ▷ User loads table from auth-protected URL (e.g. `access_url` from DataLink table)
    - Has to establish auth context as part of load activity
    - No additional info about where to find auth metadata (/capabilities)
  - ▷ ("Simple" services Cone, SIA, SSA could fit into either scenario)
- Nice to have:
  - ▷ Auth confirmation from service

```
X-VO-Authenticated: <authenticated-user-id>
```

Other VO clients are available

- and they may have different requirements

Context

- SSO 2.0/TAP 1.1
  - Lists available authentication options:
    - ▷ `<securityMethod>` elements in in /capabilities
  - But doesn't tell clients how or where to log in/acquire credentials
- Pat's strawman May 2020 (PDF):
  - Suggests an answer:

    convey login instructions in `WWW-Authenticate` **header** (RFC 7235)
    - ▷ Header contains one or more scheme-specific *challenges*
    - ▷ Header+challenge(s) must be present with **401 Unauthorized** response
    - ▷ Header+challenge(s) may be present with other responses (including **200 OK**)
  - some non-standard auth schemes/parameters required
    - ▷ apparently permitted by relevant standards

Challenge Details

Challenge syntax

- RFC 7235:
  - General format:

    `WWW-Authenticate: <auth-scheme-name> <scheme-params>`
  - Example (Basic Auth, RFC 7617):

    `WWW-Authenticate: Basic realm="WallyWorld"`
- `<auth-scheme-name>` must match RFC7230 *token* syntax
  - "`ivo://ivoa.net/std/SSO#cookie`" is not syntactically legal (contains "/")
  - Could use:
    - `WWW-Authenticate: vo-sso-cookie loginurl="https://..."`
    - `WWW-Authenticate: vo-sso securityMethod="ivo://ivoa.net/std/SSO#cookie" loginurl="https://..."`
    - ... or something else
  - Easy to solve — just choose one
  - There is an IANA registry of these `<auth-scheme-name>`s, but registration not required

Challenge specifics:

- Define `<scheme-params>` for cookie login protocol
- + similar questions for other SSO auth mechanisms that we want to use

Bootstrapping

How to establish auth context for TAP-like service?

- Get login instructions from /capabilities endpoint (new `<securityMethod>` extensions)
  - Preferred by Gaia/ESA?
  - Duplicates challenge-based login instructions, more standardisation required
- Wait for challenge with 401 during normal service interaction?
  - User may find out they are unauthenticated at late stage (e.g. after entering ADQL)
  - Doesn't work for services that work in both Anon + Auth mode (401 is never encountered)
- Get challenge some other way using existing endpoints?
  - Provoke 401 with dummy request `WWW-Authenticate: tell-me-how`?
  - All service responses (including 200s) include `WWW-Authenticate` challenges?
  - Either would work if services cooperate, but what endpoint to use?
    - `/capabilities`? `/tables`? Might have different auth than query endpoints
    - TAP `/sync`/`/async` query endpoint? might fail for non-auth reasons
- Dedicated authentication endpoint? (Markus suggestion):
  - Dedicated endpoint just to issue `WWW-Authenticate` challenges
  - Response is 200 for auth/anon service, 401/403 for auth-only service
  - DaCHS has prototyped this as `<tap-base-url>/authcheck` endpoint
    - Seems to work nicely

Client Implementation

TOPCAT/STILTS client auth implementation

- Prototype AUTH library
  - Java standalone client library (minimal dependencies)
  - To be used by TOPCAT/STILTS
  - Could be used by other java clients
  - Various backends prototyped:
    - Cookie (GACS-flavour)
    - Cookie (generic)
    - HTTP Basic Auth
    - Extensible — should(?) be easy to add new ones
  - Still under development
  - Looking for more services/standards to implement against
    - Parts of it are working, but no services have enough functionality for full test
- (builds on various discarded earlier attempts)

AUTH Library Usage

Authentication managed by (application-wide) AuthManager instance

- Basic usage:

```
AuthManager authManager = AuthManager.getInstance();
authManager.setUserInterface(UserInterface.GUI);  // or CLI
...
InputStream urlIn = authManager.connect(resourceUrl);
```

- All connections which *may* be auth-protected should go via AuthManager
- 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)
- Does not examine /capabilities for `<securityMethod>`s
- But can be primed by pointing at an `/authcheck`-like endpoint

Is it a good design?

- Can't tell yet — depends on how services/standards shape up

Service Implementation

Services I know about:

- DaCHS (GAVO)
  - `/authcheck` endpoint to bootstrap auth context
  - Anon/BasicAA TAP service
  - But not more complicated auth schemes (e.g. cookie)
- GACS (ESA/Gaia)
  - WWW-Authenticate cookie challenge — working
  - But currently no way to bootstrap auth on auth+anon endpoint
- CADC ($\rightarrow$ LSST?)
  - coming real soon now
- ... more?

Summary

Status

- WWW-Authenticate challenges (RFC7235) for login info seems to work
  - Some additional work on the specifics required
- Bootstrapping authentication for TAP-like service still problematic
  - Suggest `/authcheck`-like dedicated endpoint to supply challenges
- Implementation
  - Java standalone client library under development (hopefully usable by third parties)
  - Partial testing done against some services
  - Would like to see more service implementations to test against
- More discussion/details: grid@ivoa.net mailing list June, July 2020