Authentication

Shih, A. Haigron, R. Le Sidaner, P.

IVOA Interop, Shanghai 14-17/05/2017

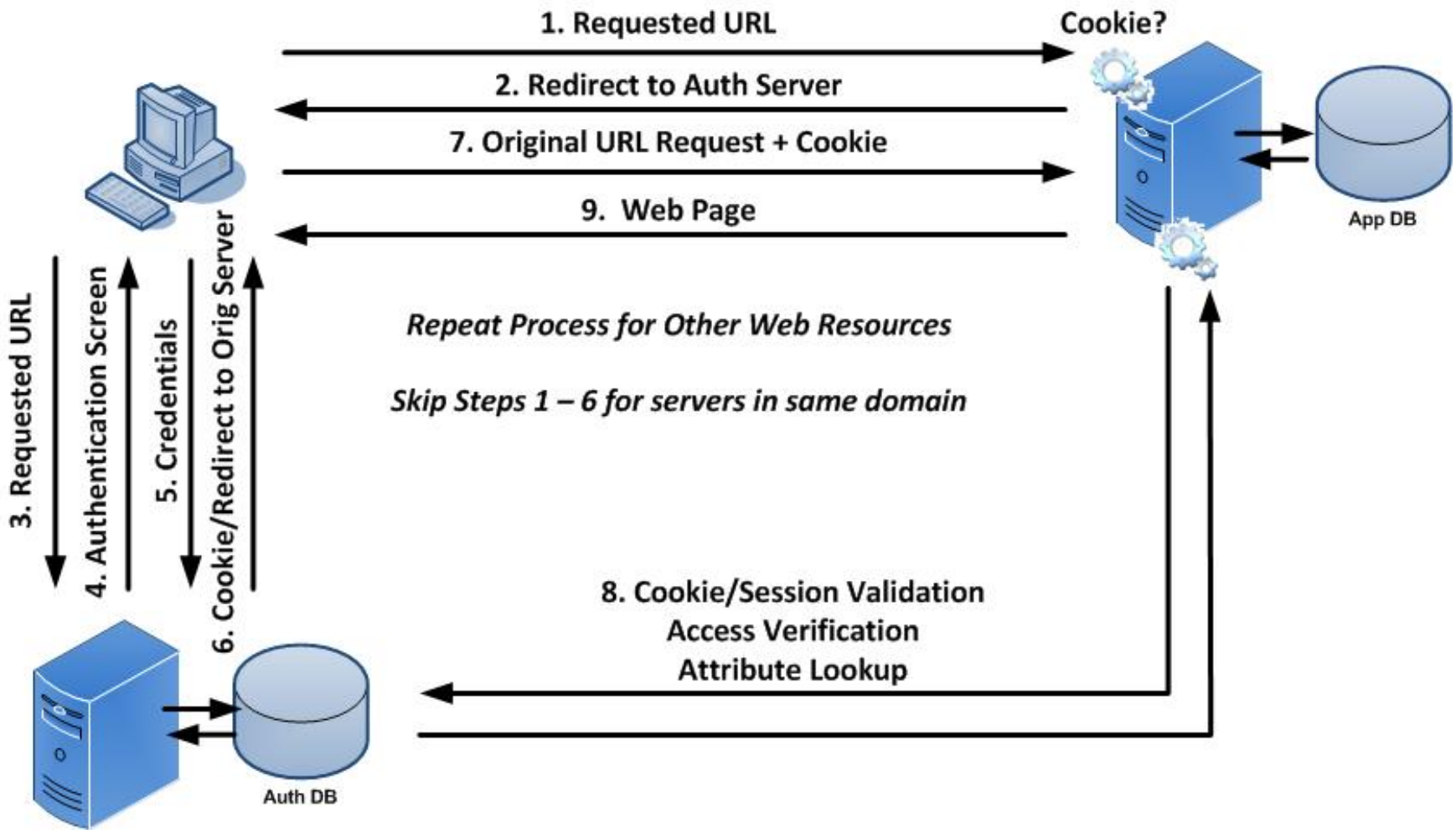
Why we need it. . . in Open Data

- For proprietary period.
- To use VO infrastructure *before* publishing.
- To use VO resources (like computing node, storage, etc.) on demand.
- To prepare publishing large project.
- etc.

What we need

- Don't want to manage people.
- Ability to authenticate non web application like `ssh/rsync also Aladin Topcat`.
- Ability to authenticate existent applications.
- Ability to manage easily authorizations.
- Easy to integrated in new applications **and old** applications.
- Easy to deploy.
- Easy to maintain with few manpower.
- Secure.

How SSO works



Problems

- Highly based on `http-redirect`, don't work well outside web-browser.
- Hard to use on CLI (`ssh`, etc.)
- Lots of implementation : SAML2 (shibboleth), oauth, openid, etc.
- Complex to very complex to integrate.
- Don't integrate authorizations, each application must manage it own authorizations, meaning each application provider must implement his own tools.

LDAP

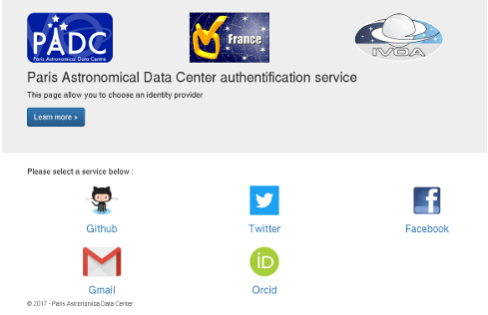
- Why
  - Use LDAP for authentication beckoned over ID Federation.
  - LDAP is well documented protocol.
  - All (almost) application can easily to use LDAP as authentication back-end.
  - Easy to use on CLI.
  - LDAP as « group » notion. Use LDAP group as authorizations back-end.
  - Easy to centralize.
- But
  - Don’t want to populate the LDAP.
  - Don’t want to manage expiration.

LDAP+SSO

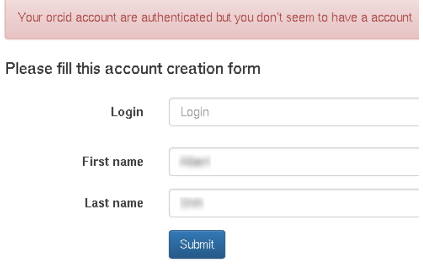
- Using SSO
- Populate a LDAP

Prototype

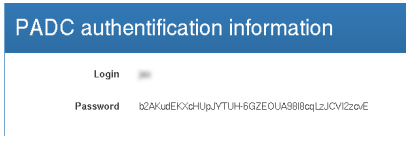
- User ask to choose a authentication service (like Orcld, Google, Github, Facebook etc.)



- If he don't have a account, we invite him to create one.



- We generate a temporary password and add it to a LDAP



Prototype

- Use this couple login/password in all your applications.
- The password is temporary same as the TTL of a cookie any web application.
- All providers can use this LDAP authentication.

Authorizations

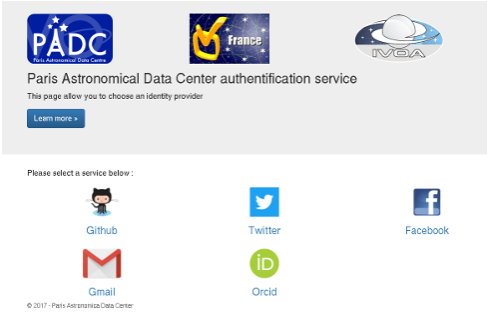
- Easy to manage authorizations
- Create group (in LDAP) like
  `cn=myapplication, ou=groups, dc=padc, dc=fr, dc=ivoa`
- Authorizations with *memberOf* test.
- For example :
  - Apache : `Require ldap-group myapplication`
  - Pam : `pam_filter |(member=cn=myapplication,ou=groups,dc=padc,dc=fr`
  - sshd : `Allowgroups` and `ldap.conf`

Using topcat and DaCHS

- We have a internal tap server (not open to all internet) `http://voparis-jpl.obspm.fr/tap`
- We don't want to modify this application.
- We put a LDAP authenticate proxy in the front of that server.

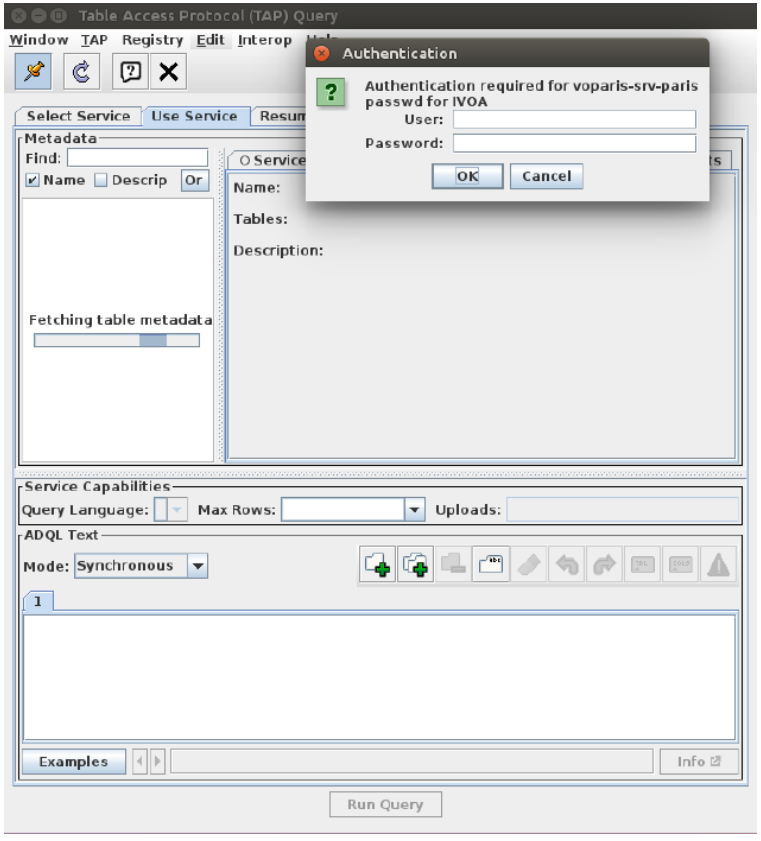
Using topcat and DaCHS

- Authentication



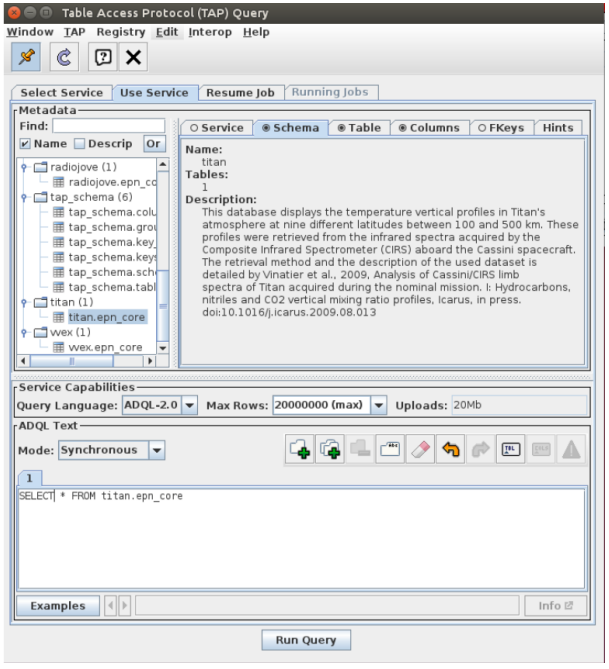
- Password

- Login in tap

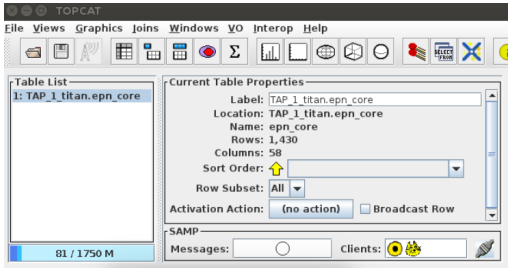


Using topcat and DaCHS

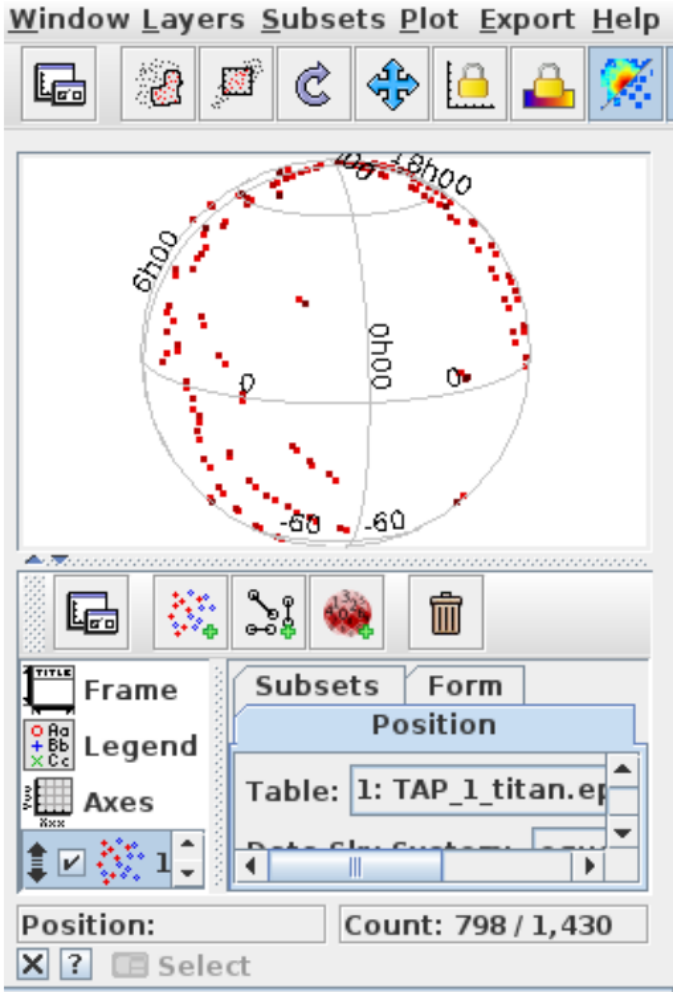
- Select



- Select



- Display



The future

- Accounts convergences :
  - Peoples who have multiple account
  - Peoples who change institution.
- Create Authorizations service.
- Delegation by branch in the LDAP.
- Delegation of the authorizations services.
- Add SAMLv2 (Shibboleth/Edugain).

Conclusion


Do you want to go for that ?

- separate Authentication between federation and application
- use LDAP as it's ready made for authentication
- have a centralised Authorisation system with delegation

Then make convergence and delegation for next interop ?