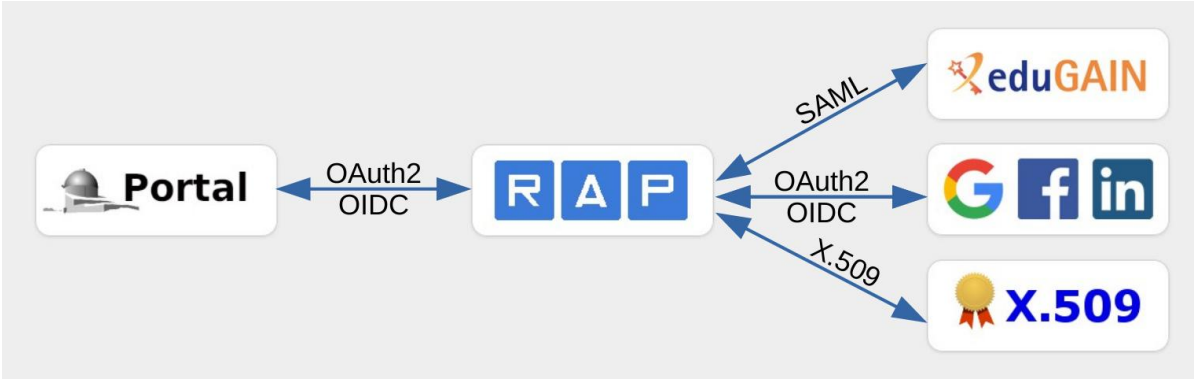
# OAuth2 / OIDC on Science Platforms

Sonia Zorba - INAF


*ADASS 2019, October 6-10, Groningen, the Netherlands*

# Use case: web access to private data




| file_name | policy |
|---|---|
| HARPN.2019-09-09T00-03-59.378.fits.gz | PRIV |
| GIANO-B.2019-09-09T00-01-52.000.fits.gz | PRIV |
| GIANO-B.2019-09-09T00-06-34.000.fits.gz | PRIV |
| GIANO-B.2019-09-09T00-11-16.000.fits.gz | PRIV |
| GIANO-B.2019-09-09T00-15-58.000.fits.gz | PRIV |
| GIANO-B.2019-09-09T00-20-41.000.fits.gz | PRIV |
| HARPN.2019-09-09T00-19-28.650.fits.gz | PRIV |

- Different protocols translated to OAuth2/OIDC
- Tokens used for communications between our services

# Use case: command line access to private data

- **Basic-Auth**

```
wget --http-user=<username> --http-password=<password> -i files_list.txt
```

- **X.509**

```
wget --certificate=<certificate-file> -i files_list.txt
```

- **OAuth2?**
  - **Step 1:** obtain a token **[...]**
  - **Step 2:** use it:

```
wget --header="Authorization: Bearer <token>" -i files_list.txt
```

  - **Step 3:** refresh the token?

# RFC 8252 - OAuth 2.0 for Native Apps

- Main issue: OAuth2 protocol is based on **redirects**, so a browser is needed in order to insert the credentials and go back to the caller application.

- Solution: redirect to a special URI:
  - listen on a port (127.0.0.1:<port>)
  - register private-use URI schemes (com.example.app://something)
  - claimed https URI

+ Proof Key for Code Exchange (PKCE)

- + keep in mind that tokens expire

**Question: should we build complex clients or use separate credentials for CLI access?**