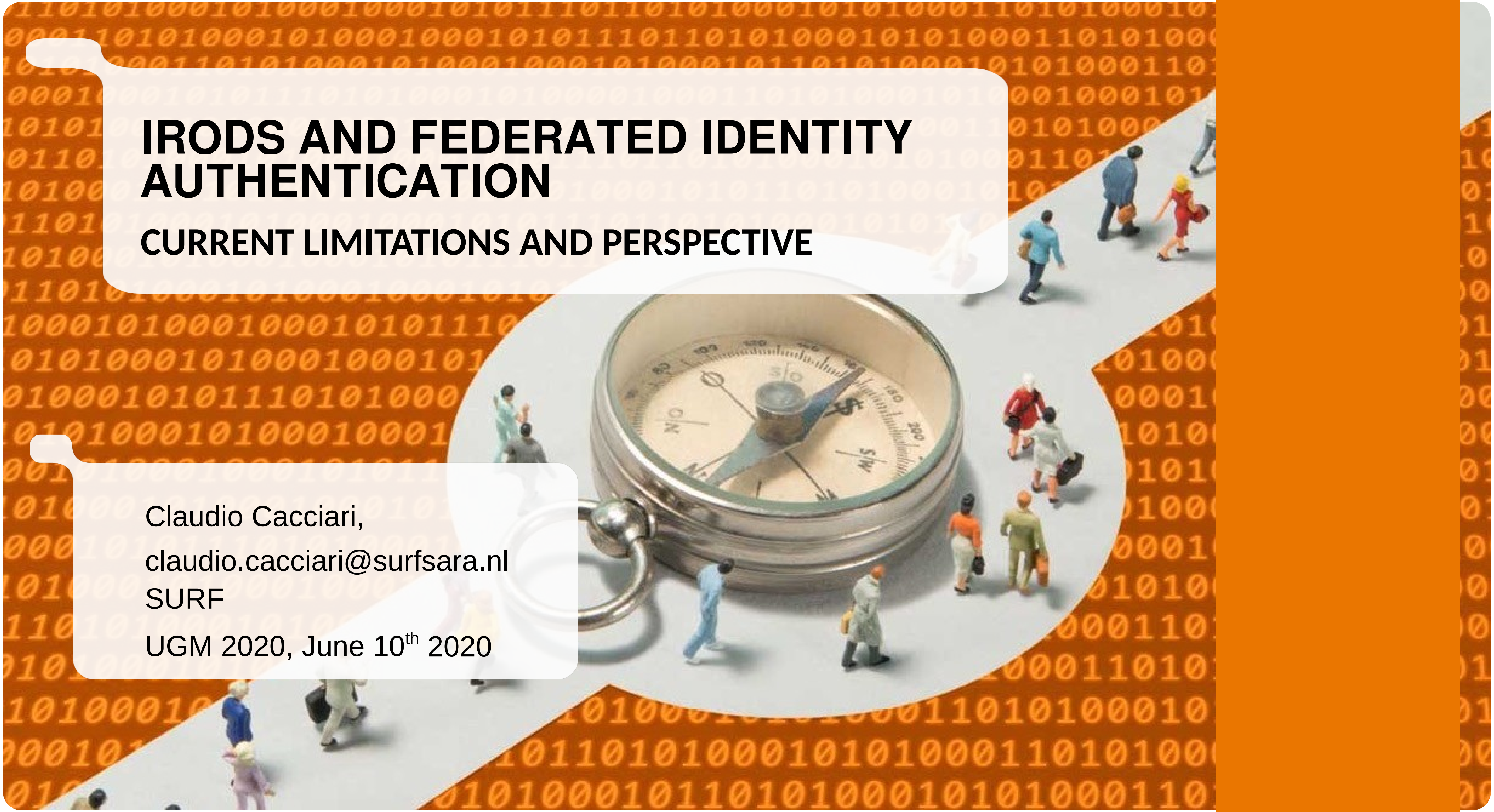
IRODS AND FEDERATED IDENTITY AUTHENTICATION

CURRENT LIMITATIONS AND PERSPECTIVE



Claudio Cacciari,

claudio.cacciari@surfsara.nl
SURF

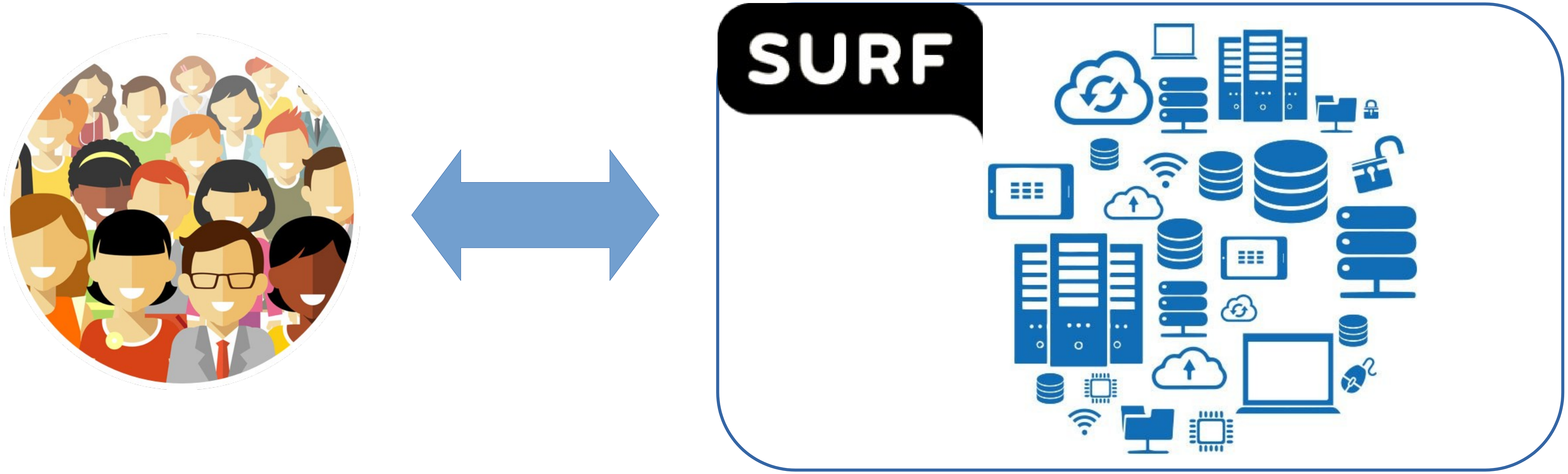
UGM 2020, June 10th 2020

IRODS and SURF

SURF is the collaborative organisation for ICT in Dutch education and research.

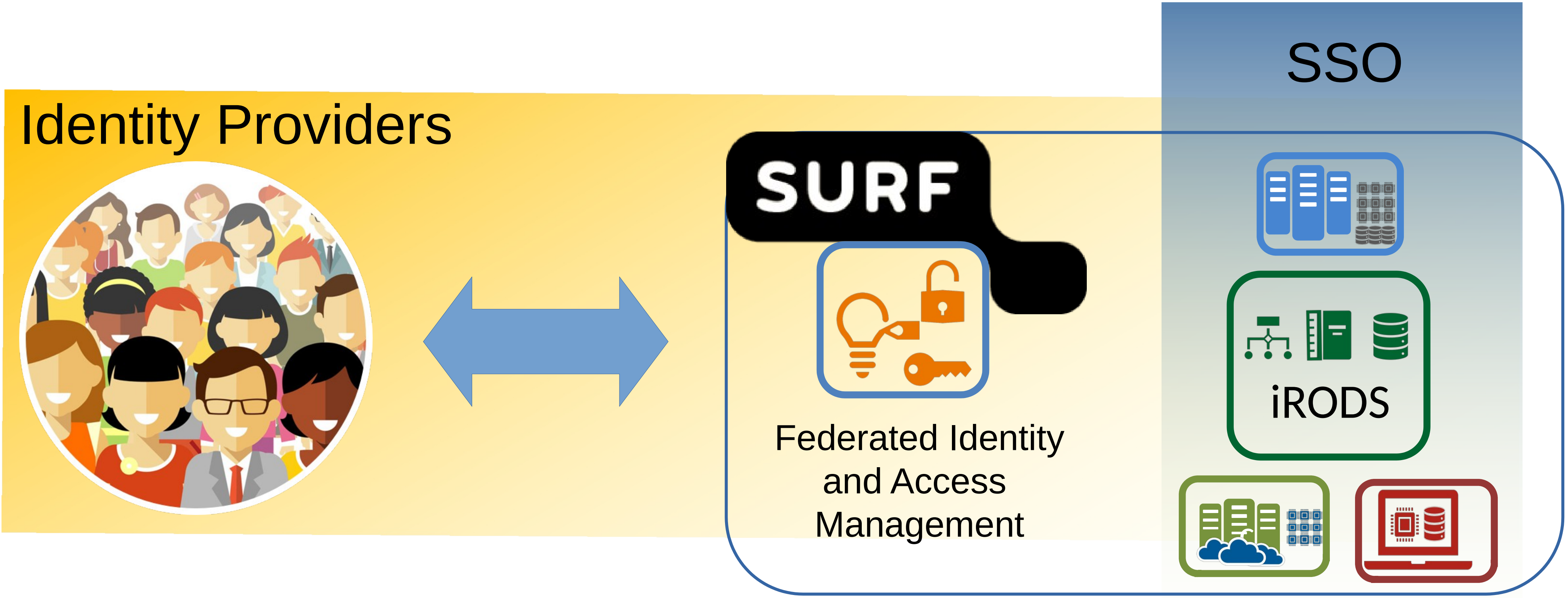
multiple organizations

multiple services



Federated Identity Management and Single Sign On

Each organization would like that their users can log in with their organization's original credentials. Some of them consider this a mandatory requirement.



Identity Providers

SSO

SURF

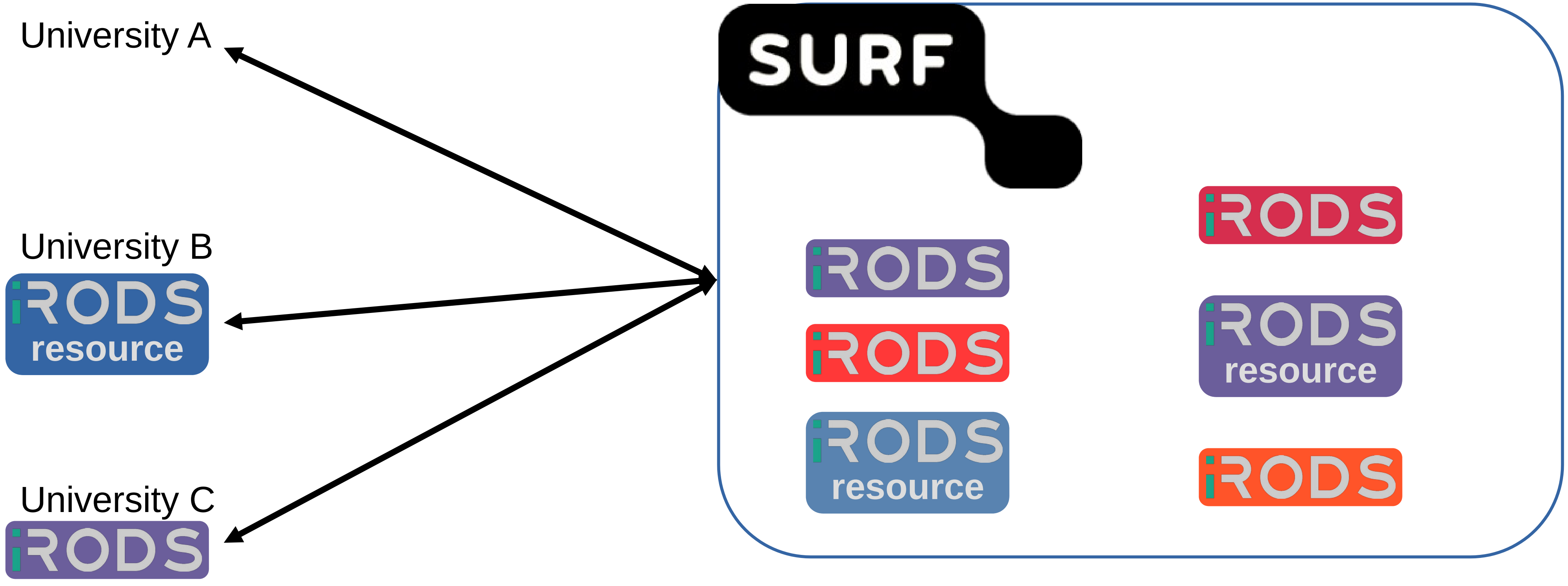
Federated Identity and Access Management

iRODS

Research Data Management services

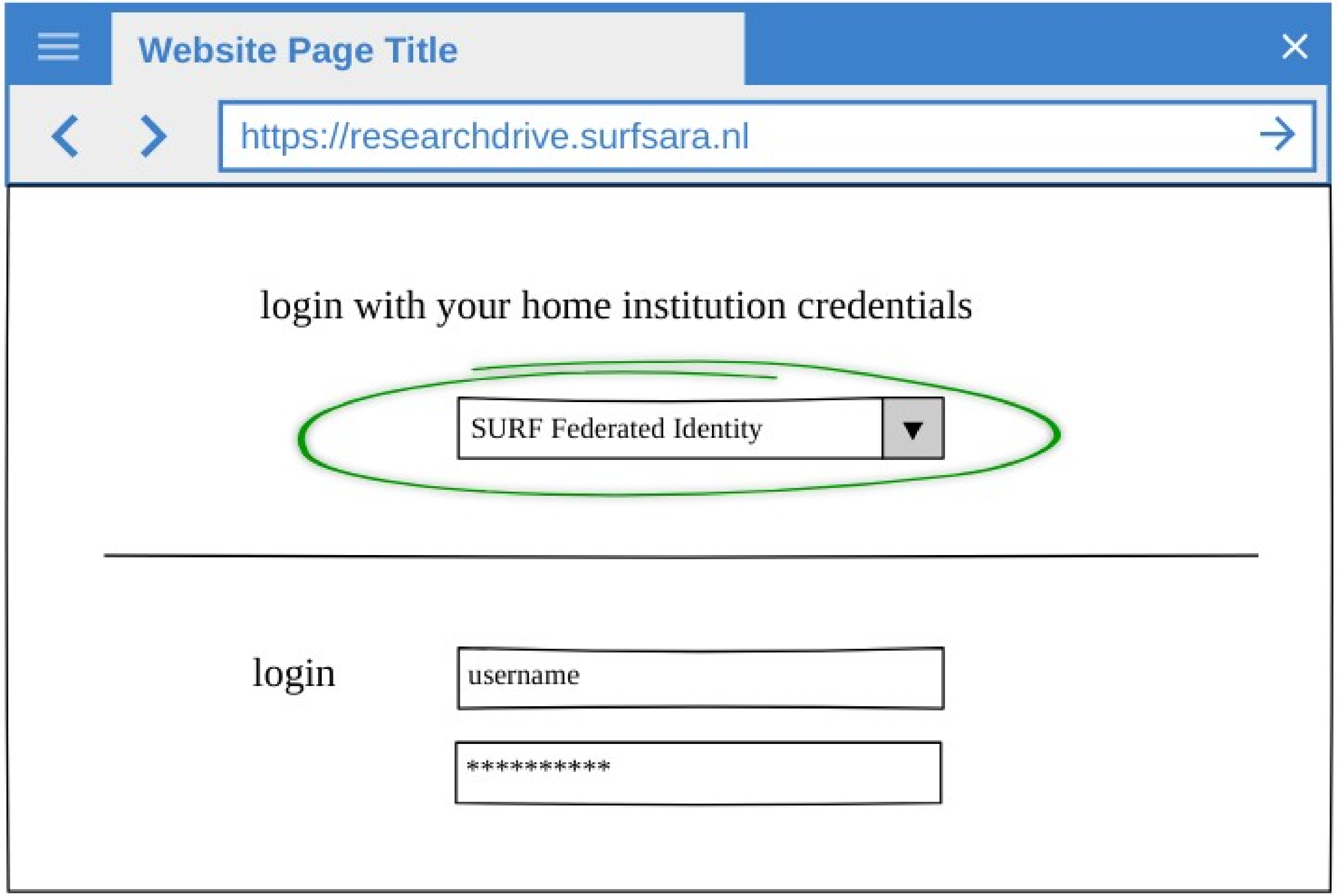
SURF offers RDM services based on iRODS:

- Each iRODS instance is dedicated for a specific organization
- Sometimes the iRODS instance is hosted by the university and SURF hosts one or more resource servers



Use case 1/5

- George, Marc and Stefanie belong to the same research team.
- Marc wants to access the data in George’s lab iRODS space through the Web interface

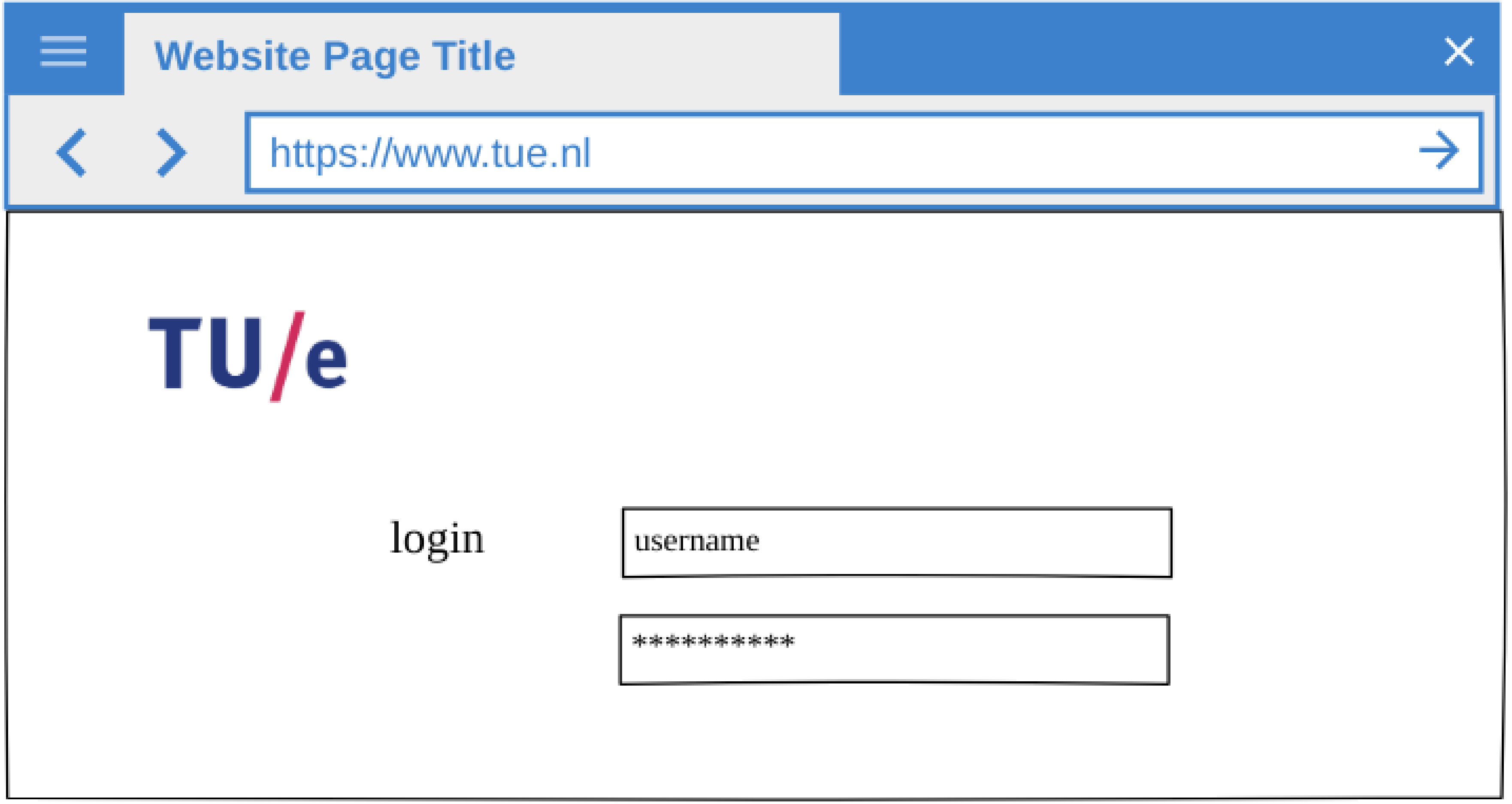


login with your home institution credentials

login

Use case 2/5

- Marc is redirected to his own institutional portal and logs in.



Website Page Title

https://www.tue.nl

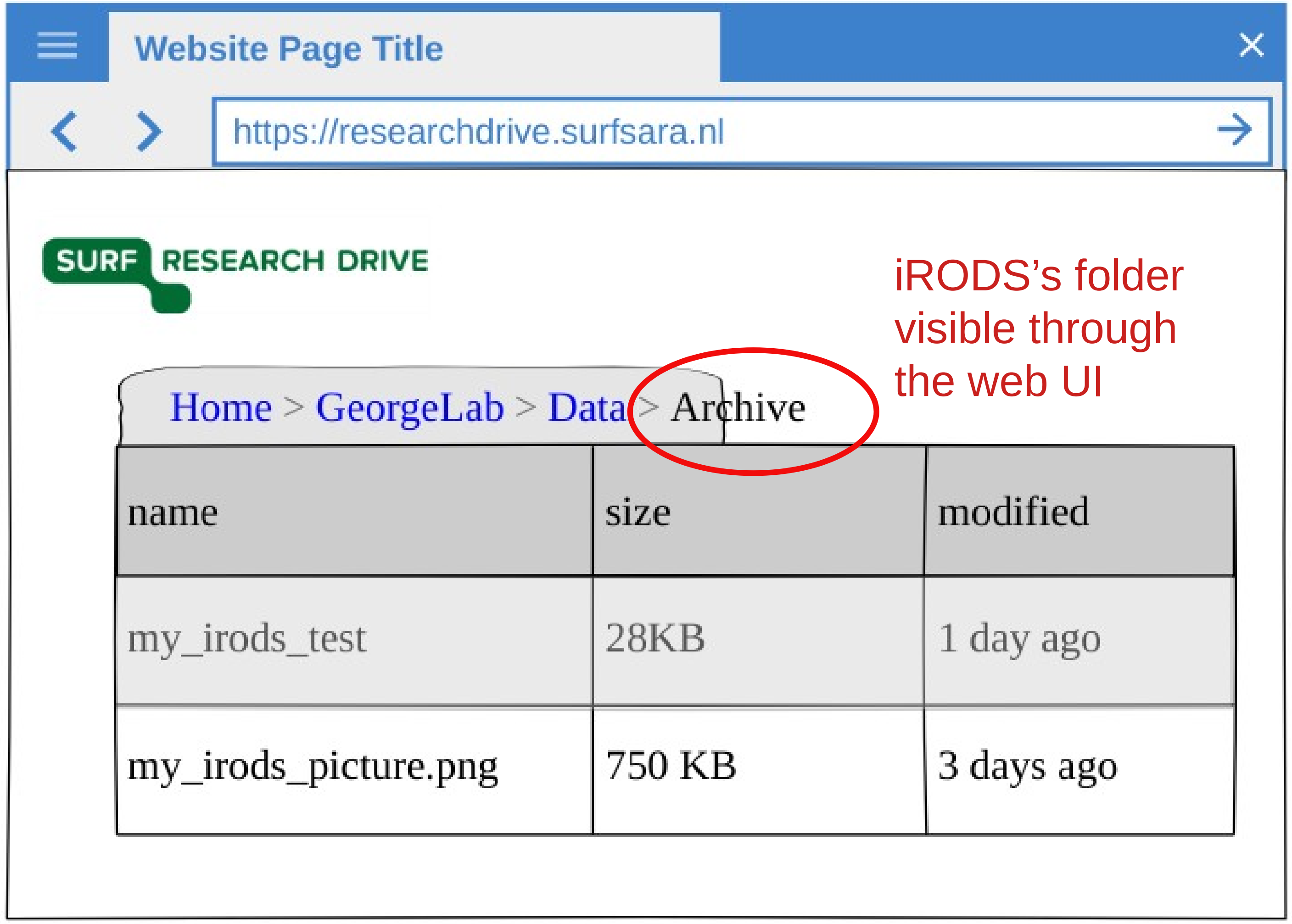
login

username

****

Use case 3/5

- Marc gets access to ~~the~~ George‘s archive folder, which is actually an iRODS folder.



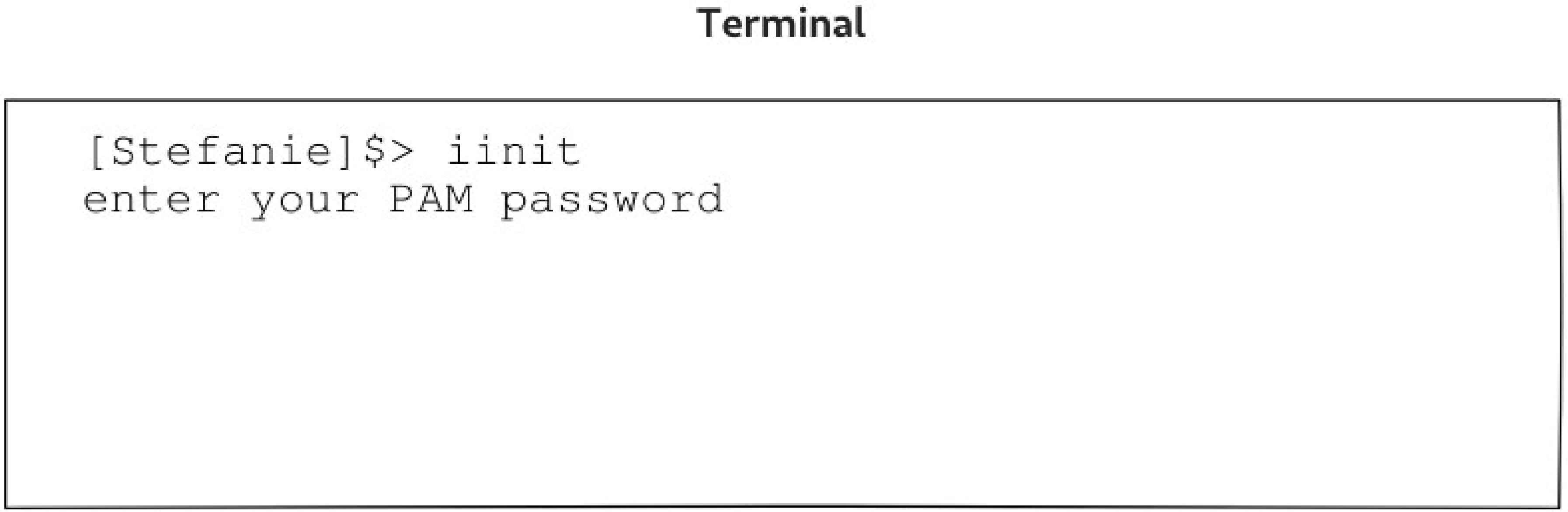
Home > GeorgeLab > Data > Archive

name	size	modified
my_irods_test	28KB	1 day ago
my_irods_picture.png	750 KB	3 days ago

iRODS’s folder visible through the web UI

Use case 4/5

- Stefanie wants to access the data in George’s lab iRODS space through the command line interface, using the identity of her institution like Marc did

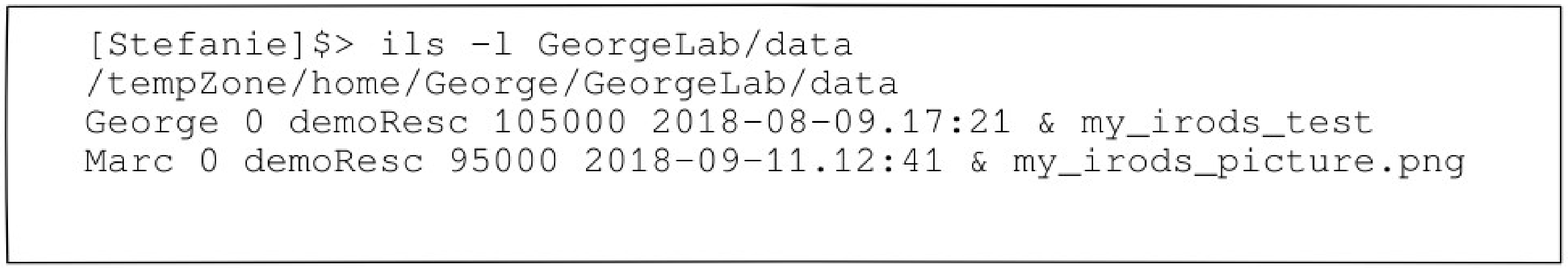


```
[Stefanie]$> iinit
enter your PAM password
```

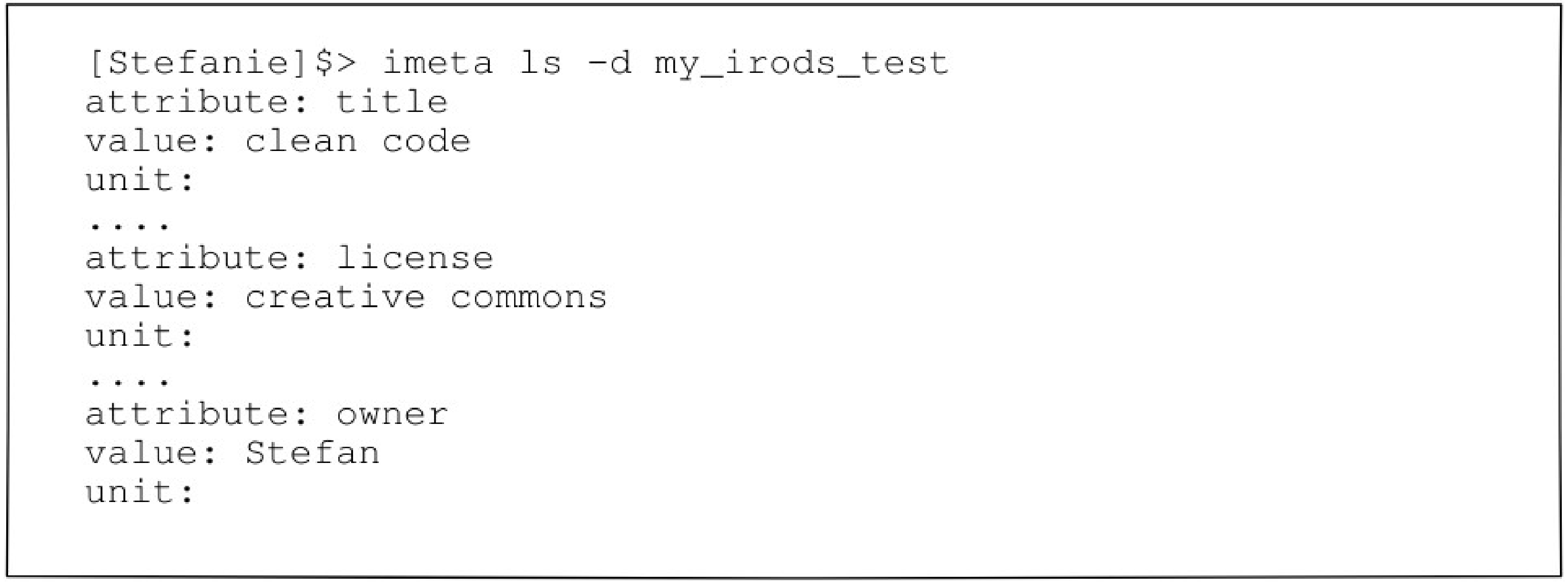
Use case 5/5

- Stefanie gets access via icommands and she can see the same data that Marc visualize through the Web UI

Terminal



```
[Stefanie]$> ils -l GeorgeLab/data
/tempZone/home/George/GeorgeLab/data
George 0 demoResc 105000 2018-08-09.17:21 & my_irods_test
Marc 0 demoResc 95000 2018-09-11.12:41 & my_irods_picture.png
```



```
[Stefanie]$> imeta ls -d my_irods_test
attribute: title
value: clean code
unit:
....
attribute: license
value: creative commons
unit:
....
attribute: owner
value: Stefan
unit:
```

iRODS behind a Web application (Marc): current solutions

In this case the user logs in the Web App and the Web App gets access to iRODS on behalf of the user.

How does the authentication work?

- it is possible to create an iRODS user for the Web App with administrator privileges
- or implementing sudo-like microservices.

- Both would be transparent for the user, like a SSO

iRODS behind a Web application: limitations

- A Web App with administrator privileges would expose the whole iRODS server if it is compromised.
- Sudo-like microservices are fine to authorize specific users for specific actions, but not for general solutions or you end up with the same issue of administrator privileges.

- None of them support FIAM
- Problematic to keep a consistent audit track

iRODS behind a Web application: token based protocols

It is possible to use OpenID Connect authentication or SAML:

- passing the token (OAuth2 access token or SAML assertion) as password in the PAM authentication plugin.
- Validating the token with a PAM OAuth2 module and mapping the “global” identity of the user to the local one.

- SAML and OIDC support FIAM and allow SSO.
- Consistent audit track.

Then solution found?!

Not yet …

iRODS behind a Web application: token based protocol limitations

- Tokens can expire.
- IRODS is not aware of the token expiration.
- Even if it were, only the Web App that has requested it, can refresh it.
- iRODS scrambled password stored at client side would outlive the token creating a security breach.

iRODS command line interface (Stefanie):

iRODS can be used as front-end via icommands:

- OIDC and SAML were not designed for command line clients.
- However with the OpenId plugin it is possible to log in via OIDC.

Problems:

- This solution does not support the approach based on access token from a Web App
- This solution requires to fix the OIDC protocol client side, limiting the flexibility of shared clients (for example Davrods)

The problem of multiple authentication protocols

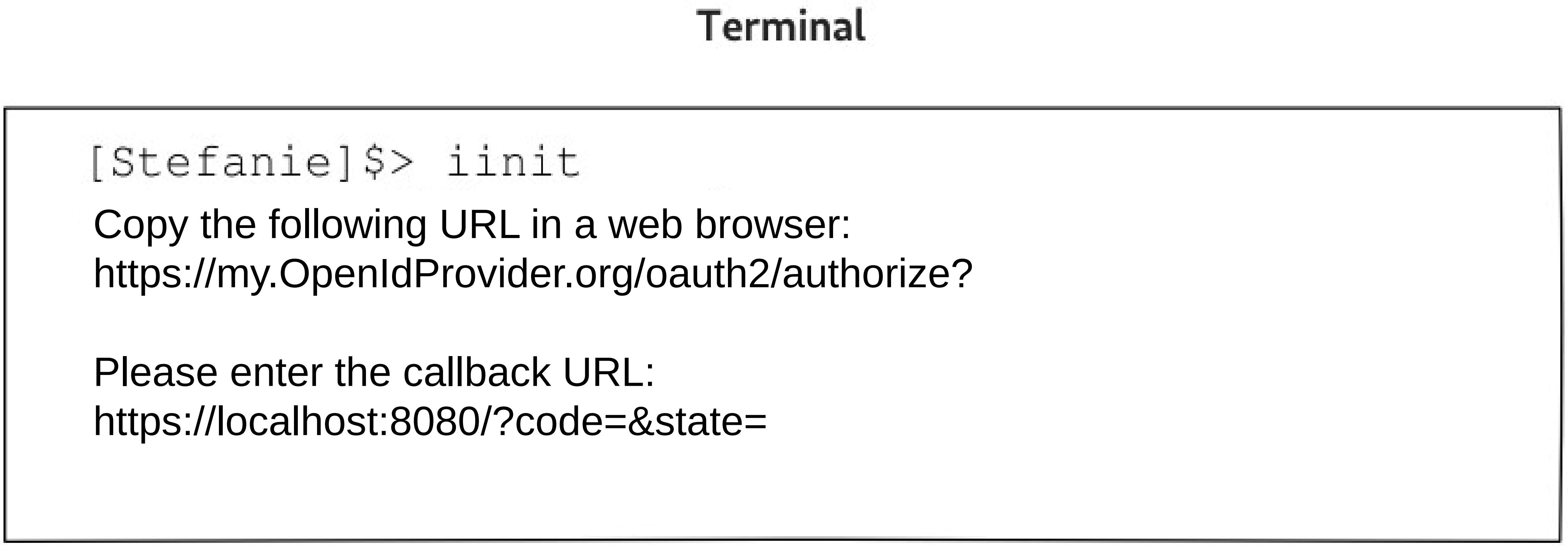
- In our environment, not all the organizations are able to provides user identities via FIM protocols. Some have just LDAP, Active Directories, Kerberos, etc.

Solution:

- PAM: pluggable authentication modules
- highly configurable: workflows defined by stack of modules
- PAM allows to define a stack of modules, each one supporting a different authentication mechanism, and the user authentication request falls through them until it is successful or reach the end of the stack.

SURF proposed solution: OIDC with PAM

- Stefanie authenticates via iinit, but using the OIDC protocol

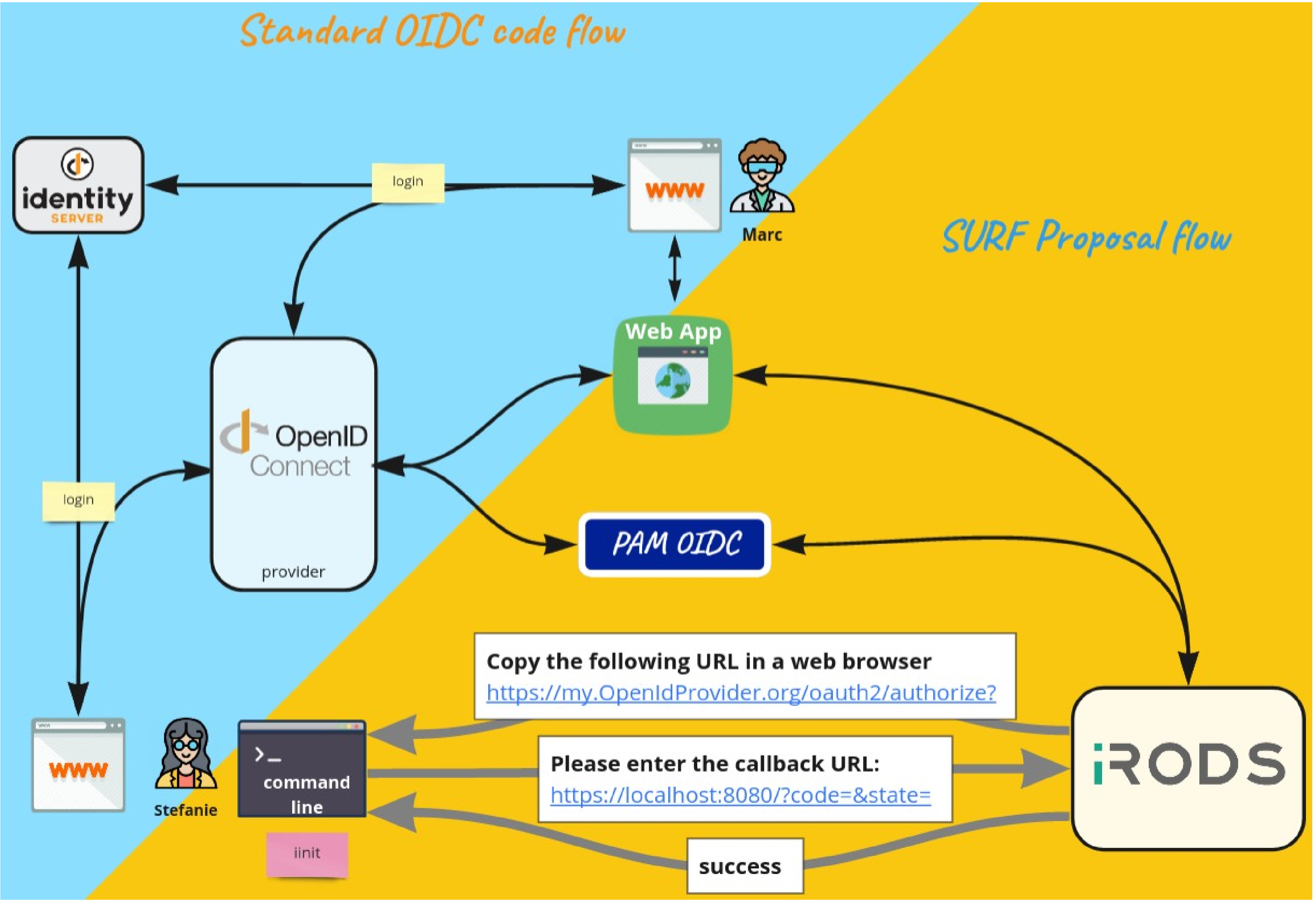


Terminal

```
[Stefanie]$> iinit
Copy the following URL in a web browser:
https://my.OpenIdProvider.org/oauth2/authorize?

Please enter the callback URL:
https://localhost:8080/?code=&state=
```

SURF proposed solution: OIDC with PAM

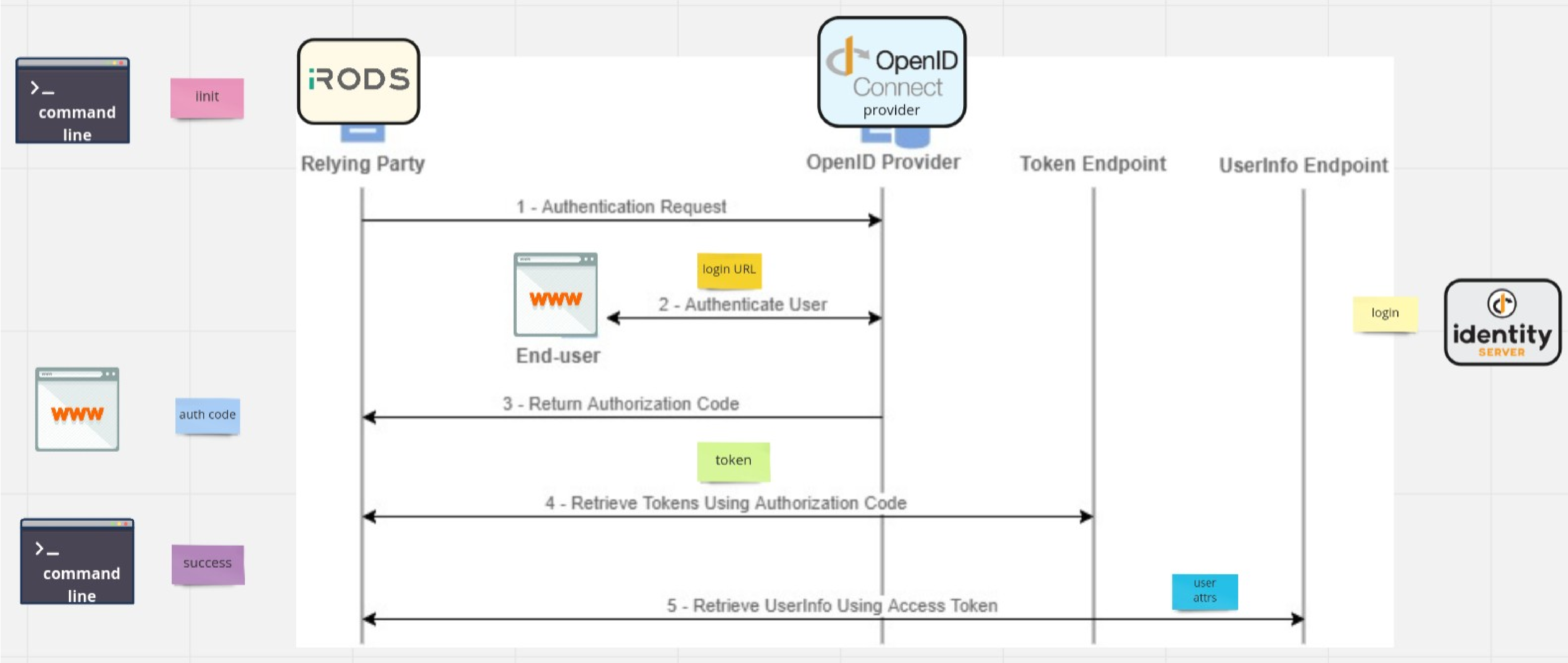


Copy the following URL in a web browser
https://my.OpenIdProvider.org/oauth2/authorize?

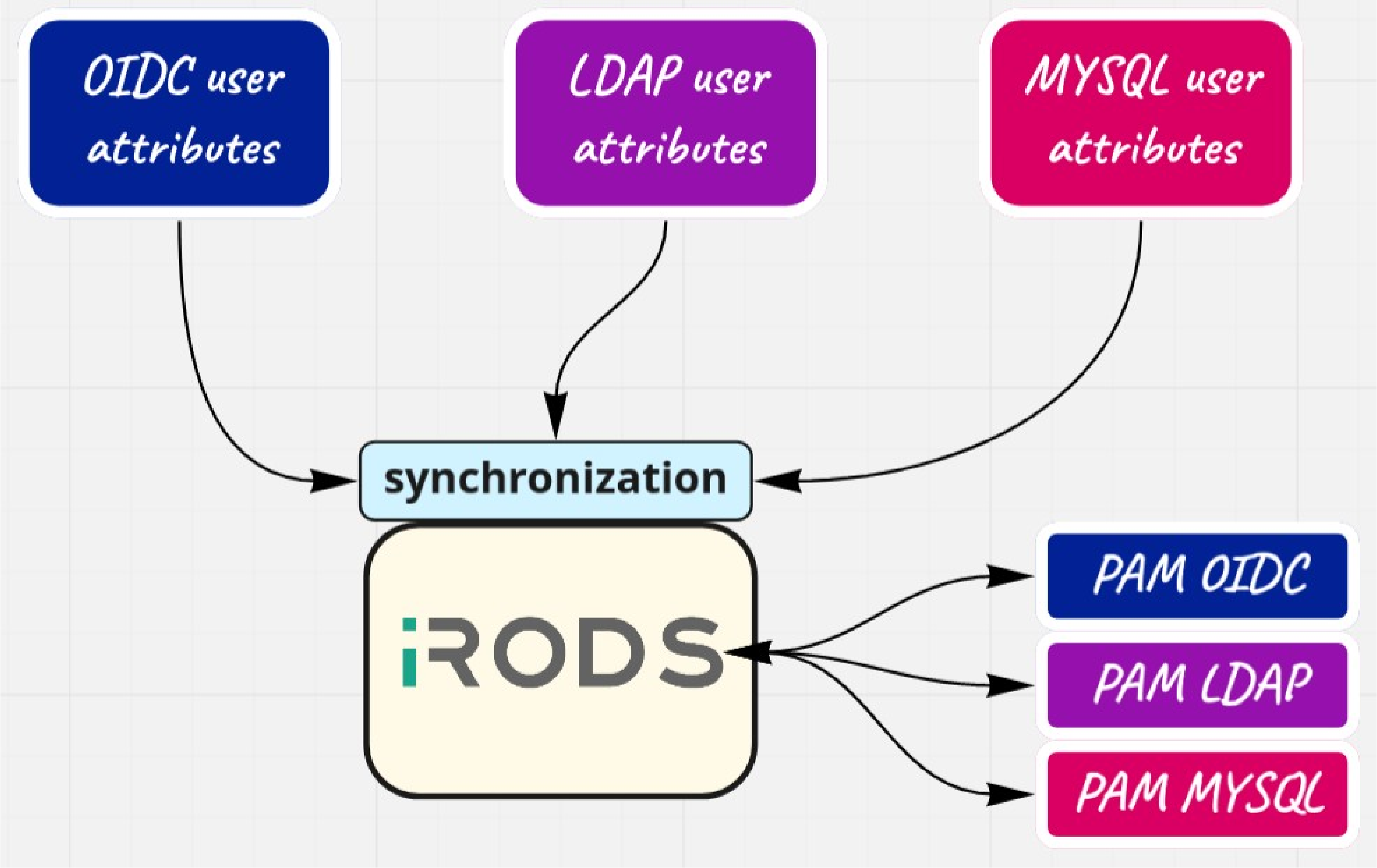
Please enter the callback URL:
https://localhost:8080/?code=&state=

success

SURF proposed solution mapped to a canonical OIDC flow



SURF proposed solution: namespace consistency



OIDC user attributes

LDAP user attributes

MYSQL user attributes

synchronization

iRODS

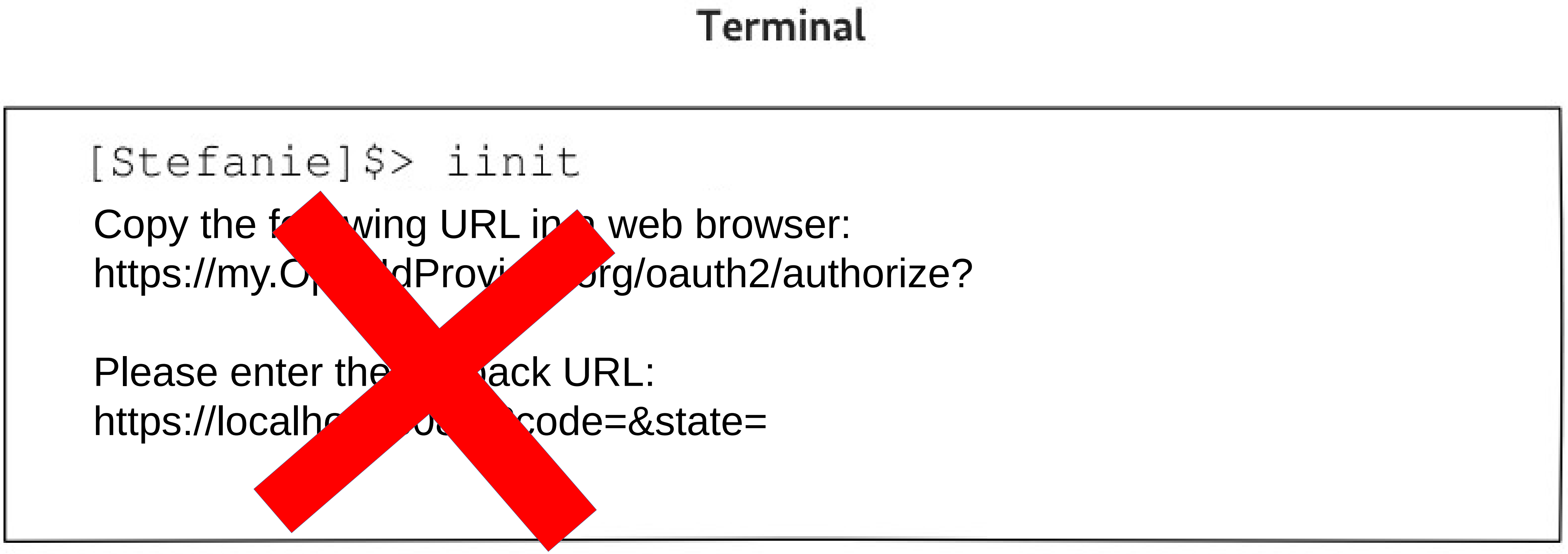
PAM OIDC

PAM LDAP

PAM MYSQL

A solution with a problem: current iRODS PAM support

- Stefanie authenticates via iinit, but using the OIDC protocol

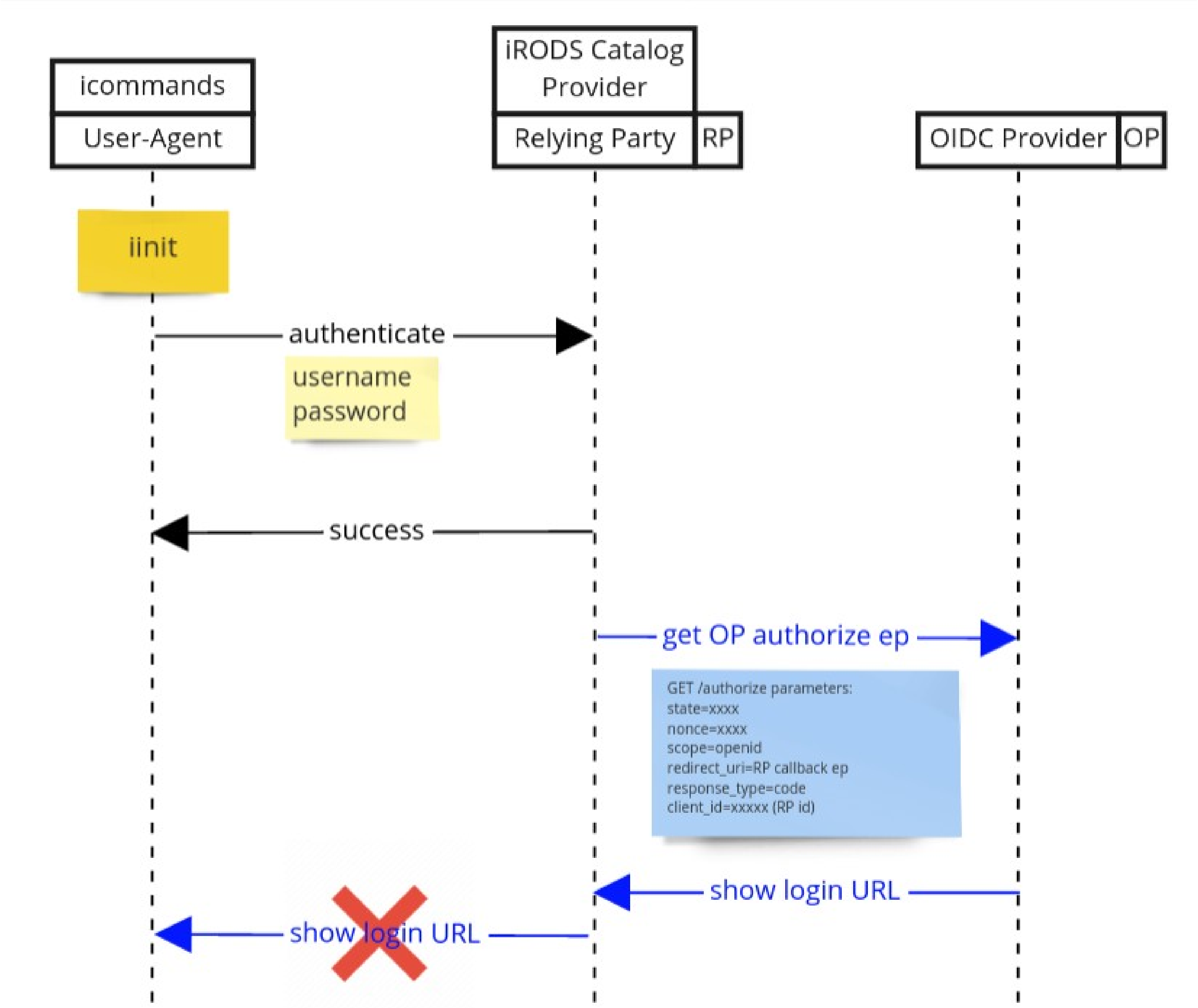


Terminal

```
[Stefanie]$> iinit
Copy the fo     wing URL in   web browser:
https://my.O       dProvi      org/oauth2/authorize?

Please enter the       ack URL:
https://localh        code=&state=
```

A solution with a problem: current iRODS PAM support

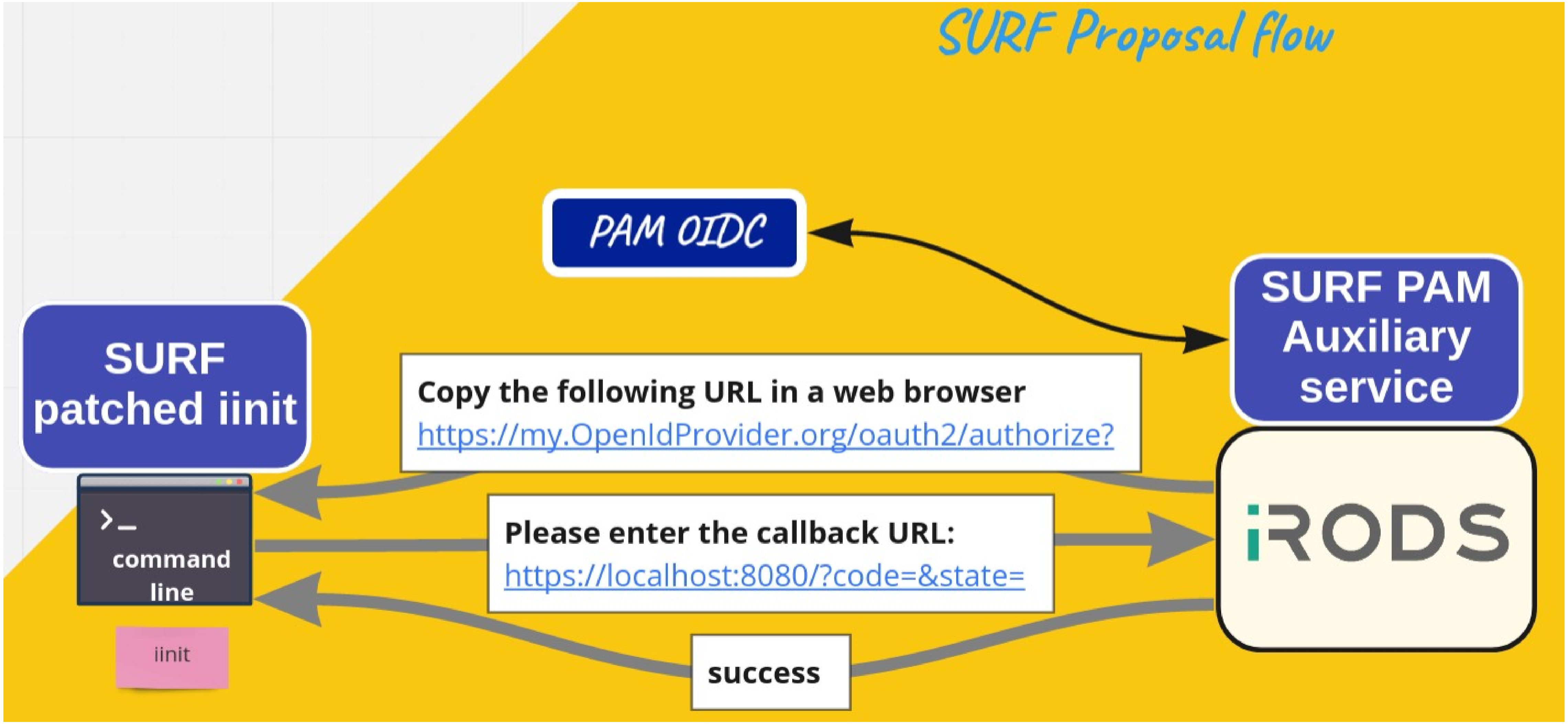


GET /authorize parameters:
state=xxxx
nonce=xxxx
scope=openid
redirect_uri=RP callback ep
response_type=code
client_id=xxxxx (RP id)

A solution with a problem: current iRODS PAM support

- The current PAM authentication plugin does not support a full PAM conversation, which is an exchange of an arbitrary number of messages between the client and server.
- The current iRODS client assumes that a scrambled password is stored locally and it is available for the other icommands, because there is no concept of a “session”.
- What should we store in case of tokens? Or other PAM modules with multiple responses?

SURF proposed solution: PAM enhanced



Copy the following URL in a web browser
https://my.OpenIdProvider.org/oauth2/authorize?

Please enter the callback URL:
https://localhost:8080/?code=&state=

success

SURF proposed solution: PAM enhanced

- It is a Proof of Concept
- It has been necessary to modify the core iRODS server and client code
- Good to start testing with our users

However

- We have no intention to maintain a patched version of iRODS
- We aim to converge towards a general solution with the iRODS consortium

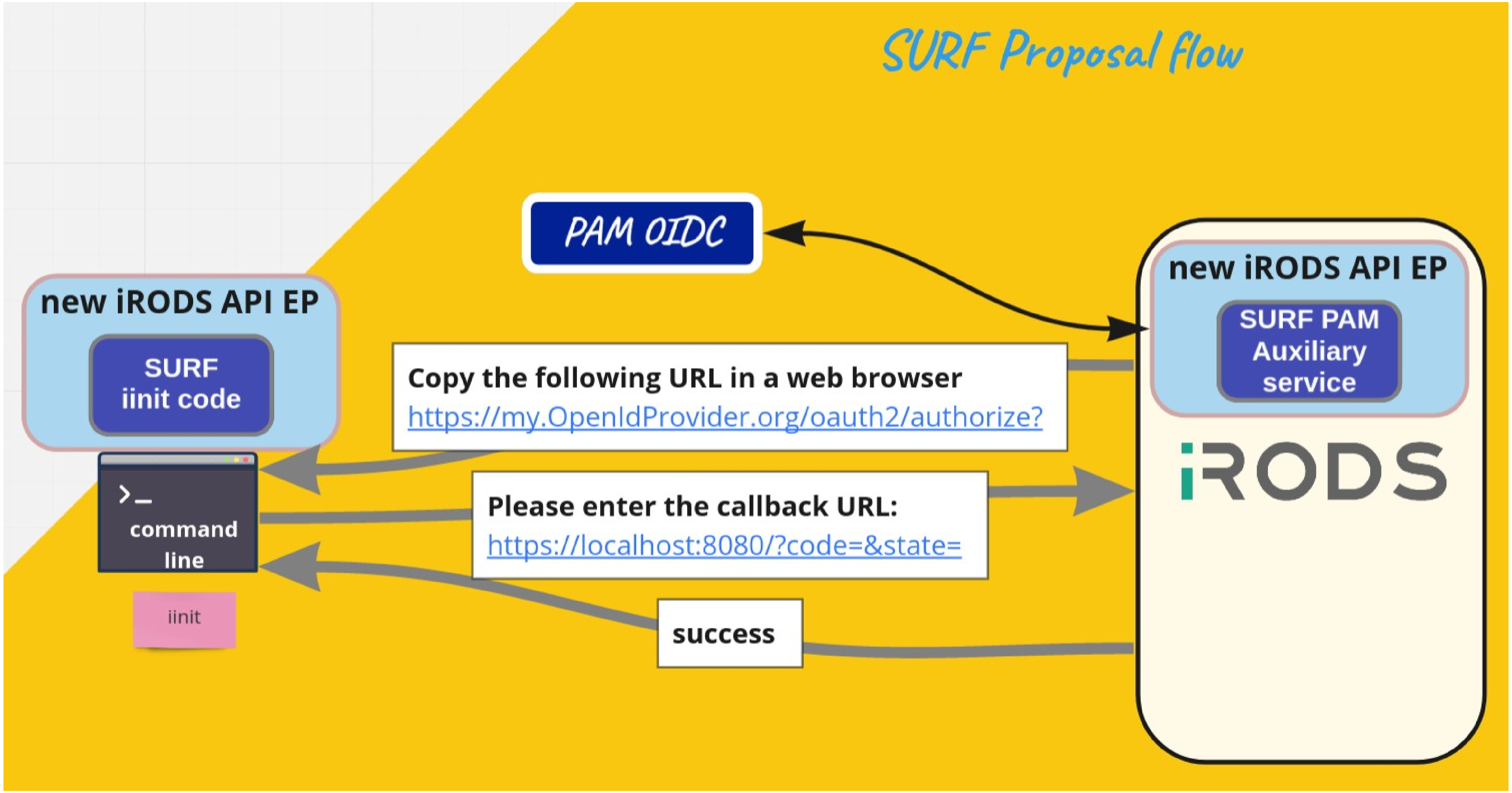
A general solution: iRODS Auth WG new API endpoint

- The iRODS Working Group:
- https://github.com/irods-contrib/irods_working_group_authentication
- proposed a new approach: defining a new iRODS API endpoint, which has the flexibility to support an arbitrary exchange of messages.

A general solution: iRODS Auth WG new API endpoint

- The messages are json documents
- A Proof of Concept has been developed by Jason Coposky
- It supports the native authentication, so to test PAM, OIDC, etc. further development is needed from the community
- How storing the responses handled by plugin still to be defined
- Since the plugin is client driven, then each client will have to be explicitly enabled: each protocol needs to be implemented in each required client library (icommands, python API, Java API, etc.)

Next steps



Copy the following URL in a web browser
https://my.OpenIdProvider.org/oauth2/authorize?

Please enter the callback URL:
https://localhost:8080/?code=&state=

success

Thanks to

- Stefan Wolfsheimer
- Hylke Koers
- Gerben Venekamp
- Arthur Newton
- Matthew Saum
- Tasneem Rahaman-Khan

Questions for you

- Do you need to connect multiple organizations to your services, using FIM?
  - Yes, no

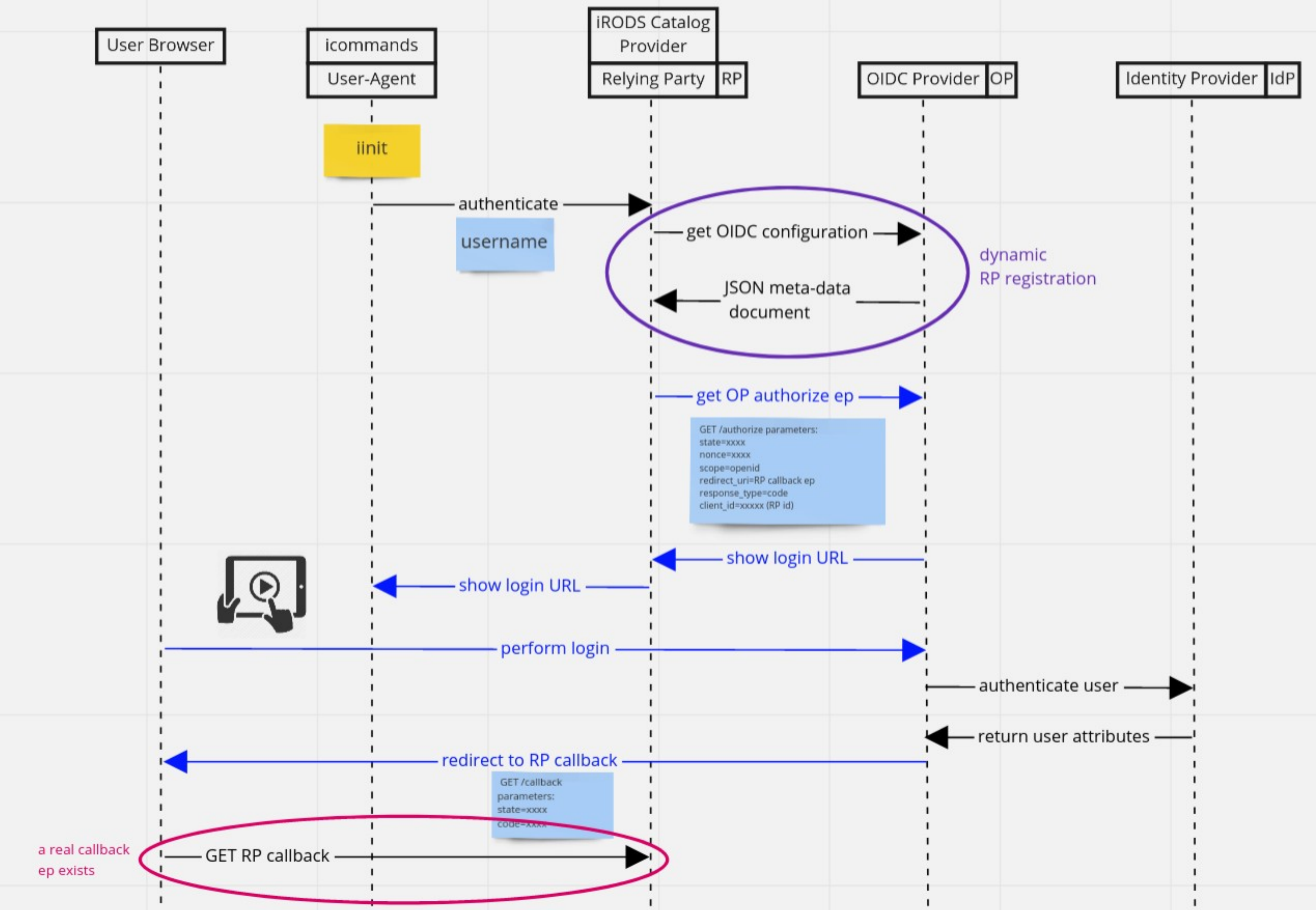
Questions for you

- Which authentication protocols do you use with iRODS?
    - OIDC, LDAP, Kerberos, Native, others

Questions for you

- Do you need to support multiple authentication protocols in each iRODS instance?
  - Yes, no

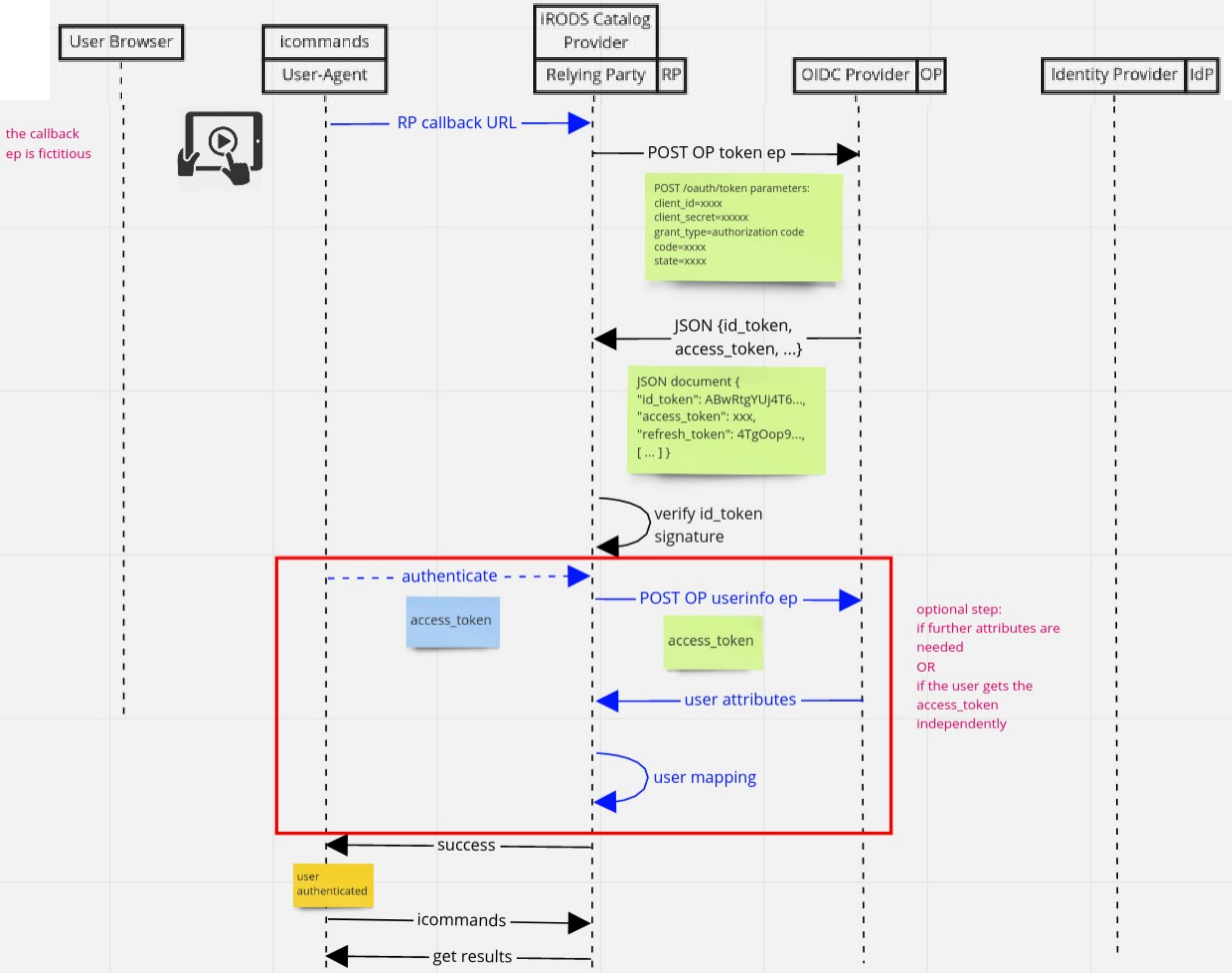
Optional slides



```
GET /authorize parameters:
state=xxxx
nonce=xxxx
scope=openid
redirect_uri=RP callback ep
response_type=code
client_id=xxxxx (RP id)
```

```
GET /callback
parameters:
state=xxxx
code=xxxx
```

Optional slides



```
POST /oauth/token parameters:
client_id=xxxx
client_secret=xxxxx
grant_type=authorization code
code=xxxx
state=xxxx
```

```
JSON document {
"id_token": ABwRtgYUj4T6...,
"access_token": xxx,
"refresh_token": 4TgOop9...,
[ ... ] }
```