# Beyond X.509: Token-based Authentication and Authorization for HEP

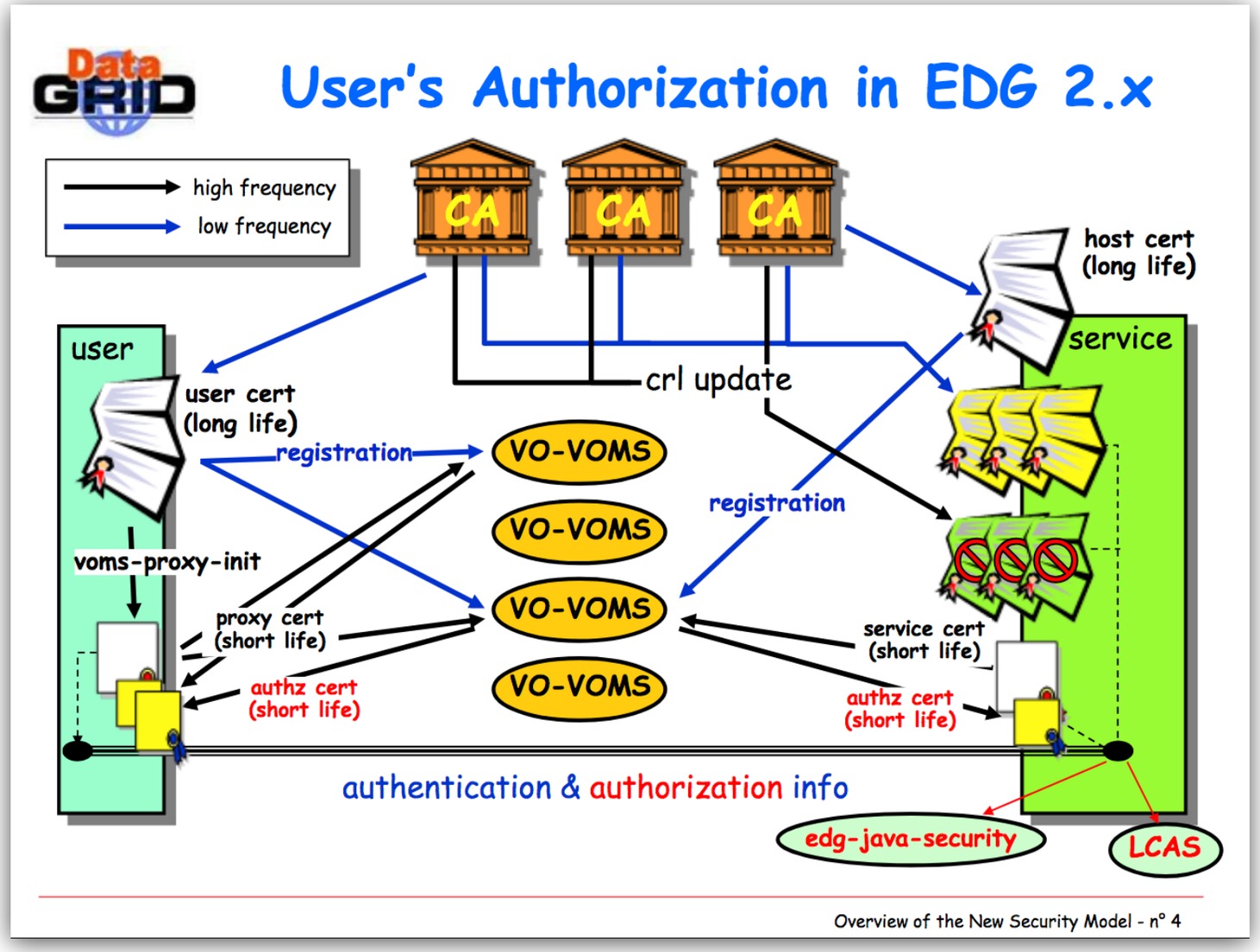
**Andrea Ceccanti**, Marco Caberletti, Enrico Vianello, Francesco Giacomini
INFN CNAF

CHEP 2018
Sofia, July 12th 2018


# The current WLCG AAI

In operation since ~2003, and still working nicely:

- **X.509 trust fabric** provided by **IGTF** (tells services which CAs are trusted)
- **X.509 certificates** provided to users for authentication
- **Proxy certificates** for Single Sign-On (SSO) and delegation
- **VOMS attribute certificates** for attribute-based authorization (issued and signed by VO-scoped VOMS servers)




Slide by Ákos Frohner

# Current WLCG AAI: the weak points

## Usability

- X.509 certificates are difficult to handle for users
- VOMS does not work in browsers

## Inflexible authentication

- Only one authentication mechanism supported: X.509 certificates
- Hard to integrate identity federations

## Authorization tightly bound to authentication mechanism

- VOMS attributes are inherently linked to an X.509 certificate subject

## Ad-hoc solution

- We had to invent our own standard and develop ad-hoc libraries and central services to implement our own AAI

## Can we do better today?

# A novel AAI for WLCG: main challenges

## Authentication

- **Flexible**, able to accomodate various authentication mechanisms
  - X.509, username & password, EduGAIN, social logins (Google, GItHub), ORCID, …

## Identity harmonization & account linking

- Harmonize multiple identities & credentials in a single account, providing a **persistent identifier**

## Authorization

- **Orthogonal** to authentication, **attribute** or **capability-based**

## Delegation

- Provide the ability for **services to act on behalf of users**
- Support for **long-running applications**

## Provisioning

- Support provisioning/de-provisioning of identities to services/relying resources

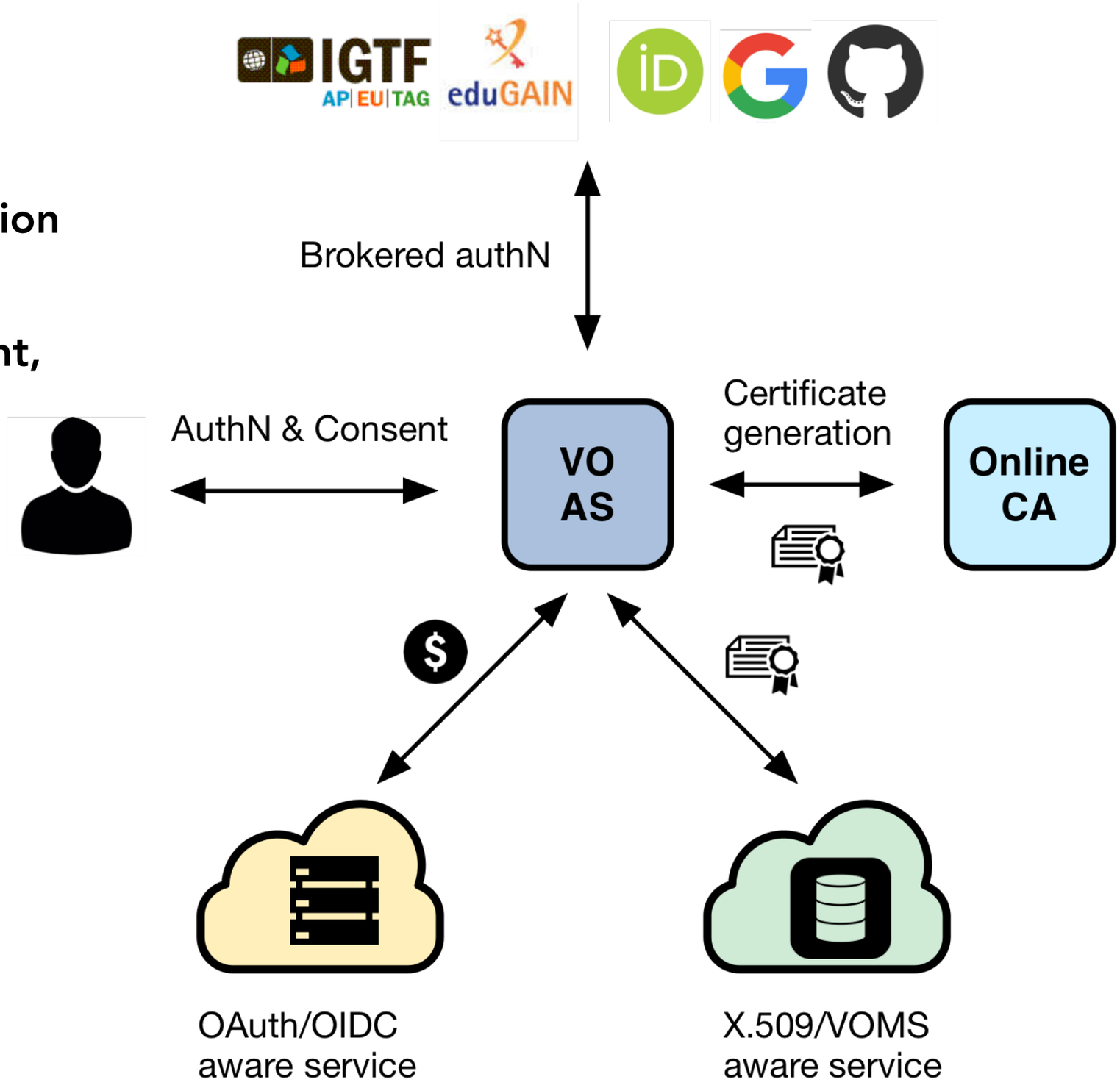
## Token translation

- Enable **integration with legacy services through controlled credential translation**

# A token-based AAI for WLCG

Introduce a central VO-scoped service that

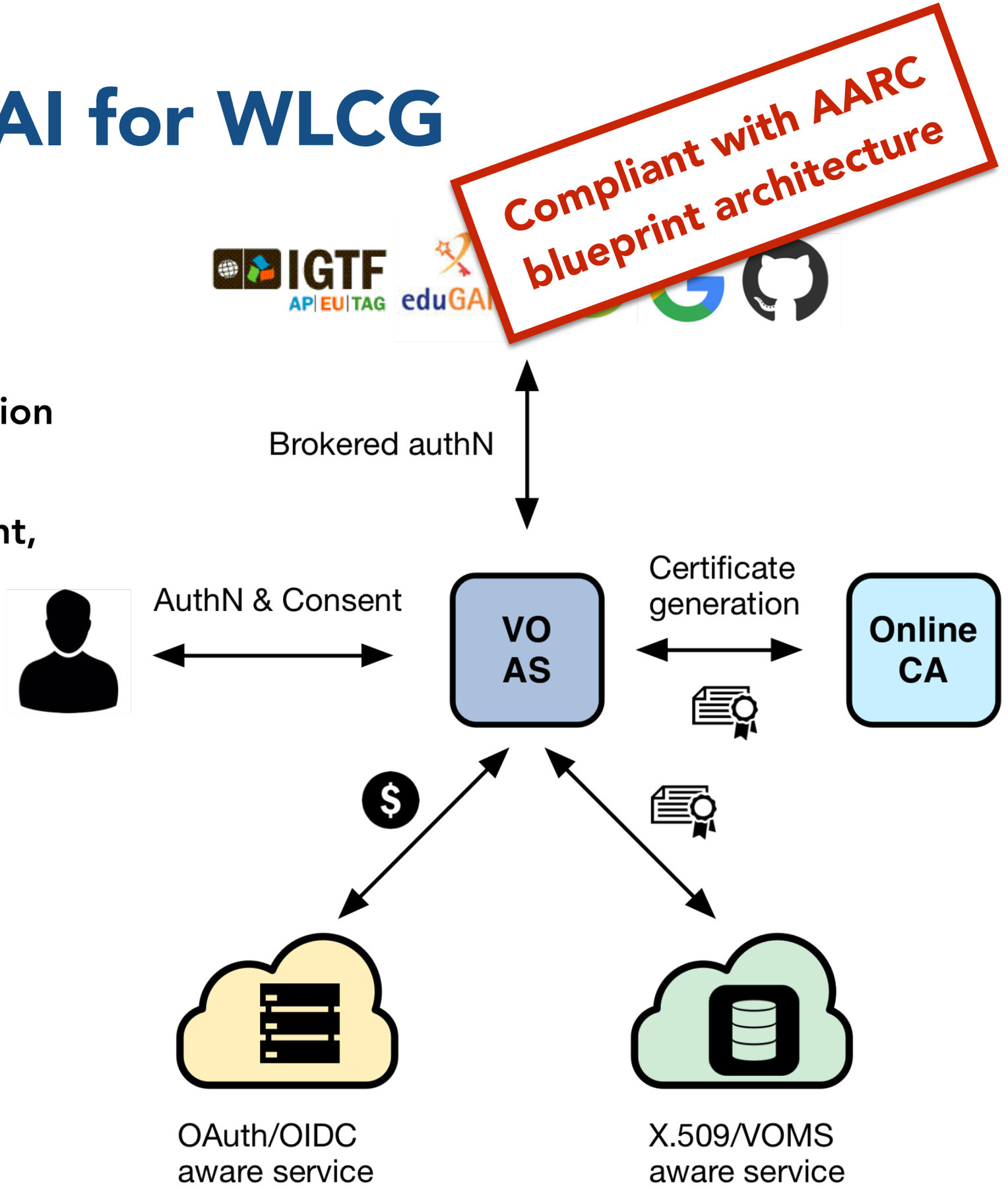
- supports **multiple authentication mechanisms**
- provides users with a **persistent, VO-scoped** identifier
- exposes **identity information**, **attributes** and **capabilities** to services via **JWT** tokens and standard **OAuth & OpenID Connect** protocols
- can integrate existing **VOMS**-aware services
- supports **Web** and **non-Web access**, **delegation** and **token renewal**



# A token-based AAI for WLCG

Introduce a central VO-scoped service that

- supports **multiple authentication mechanisms**
- provides users with a **persistent, VO-scoped** identifier
- exposes **identity information**, **attributes** and **capabilities** to services via **JWT** tokens and standard **OAuth & OpenID Connect** protocols
- can integrate existing **VOMS**-aware services
- supports **Web** and **non-Web access**, **delegation** and **token renewal**



# Enabling technologies: an overview

# Enabling technologies in one slide

## OAuth 2.0

- a standard framework for **delegated authorization**
- widely adopted in industry

## OpenID Connect

- an **identity layer** built on top of OAuth 2
- “OAuth-based authentication done right”


## JSON Web Tokens (JWTs)

- a **compact**, **URL-safe** means of representing **claims** to be transferred between two (or more) parties

# OAuth: an example delegated authorization flow

**Authorization Server**

**Resource Server**



**Resource Owner**

**Client**

**Example:**

Link a user Twitter account to his Facebook account, so that when he tweets something, the tweets are visible for his Facebook friends

# OAuth: an example delegated authorization flow

**Authorization Server**

**Resource Server**

**Resource Owner**

**Resource owner:**

The user who owns his Facebook page and drives the delegated authorization process

**Client**

# OAuth: an example delegated authorization flow

**Authorization Server**

**Resource Server**

**Resource Owner**


**Client:**

Twitter, which will be granted permission to post on Facebook on behalf of the user when the user tweets something

# OAuth: an example delegated authorization flow

**Authorization Server**
**Resource Server**



**Resource Owner**

**Client**

## Authorization Server:

Facebook AS will authenticate the user and issue tokens to the client (Twitter) after having obtained a consent from the Resource Owner

## Resource Server:

Facebook RS (APIs) will grant access only to those clients presenting a valid token, and limit actions according to the scopes linked to the token

# OAuth: an example delegated authorization flow

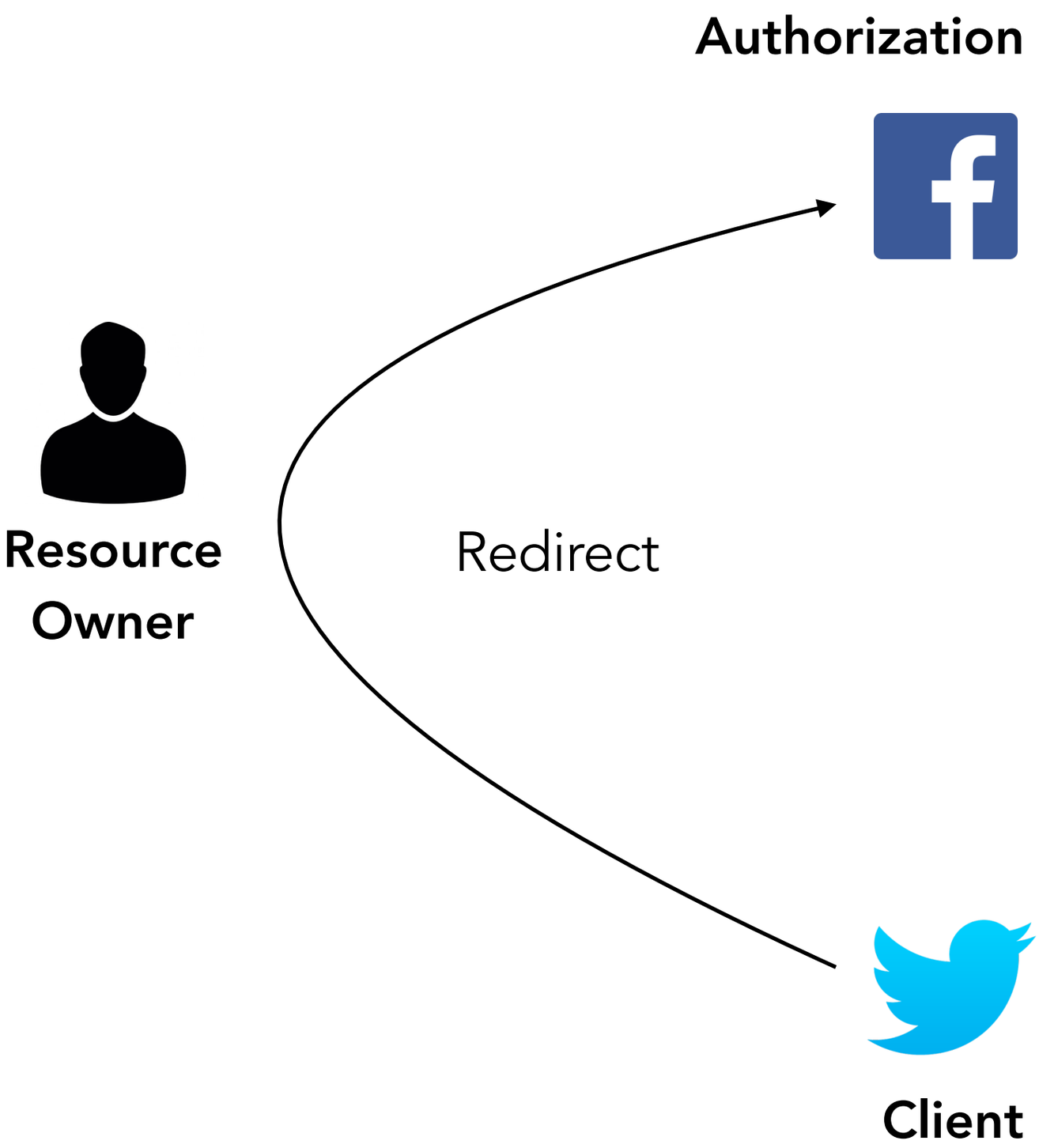
**Resource Owner**

Connect to my Facebook account

**Client**

The user (aka Resource Owner) requests that his Twitter account is connected to his Facebook page

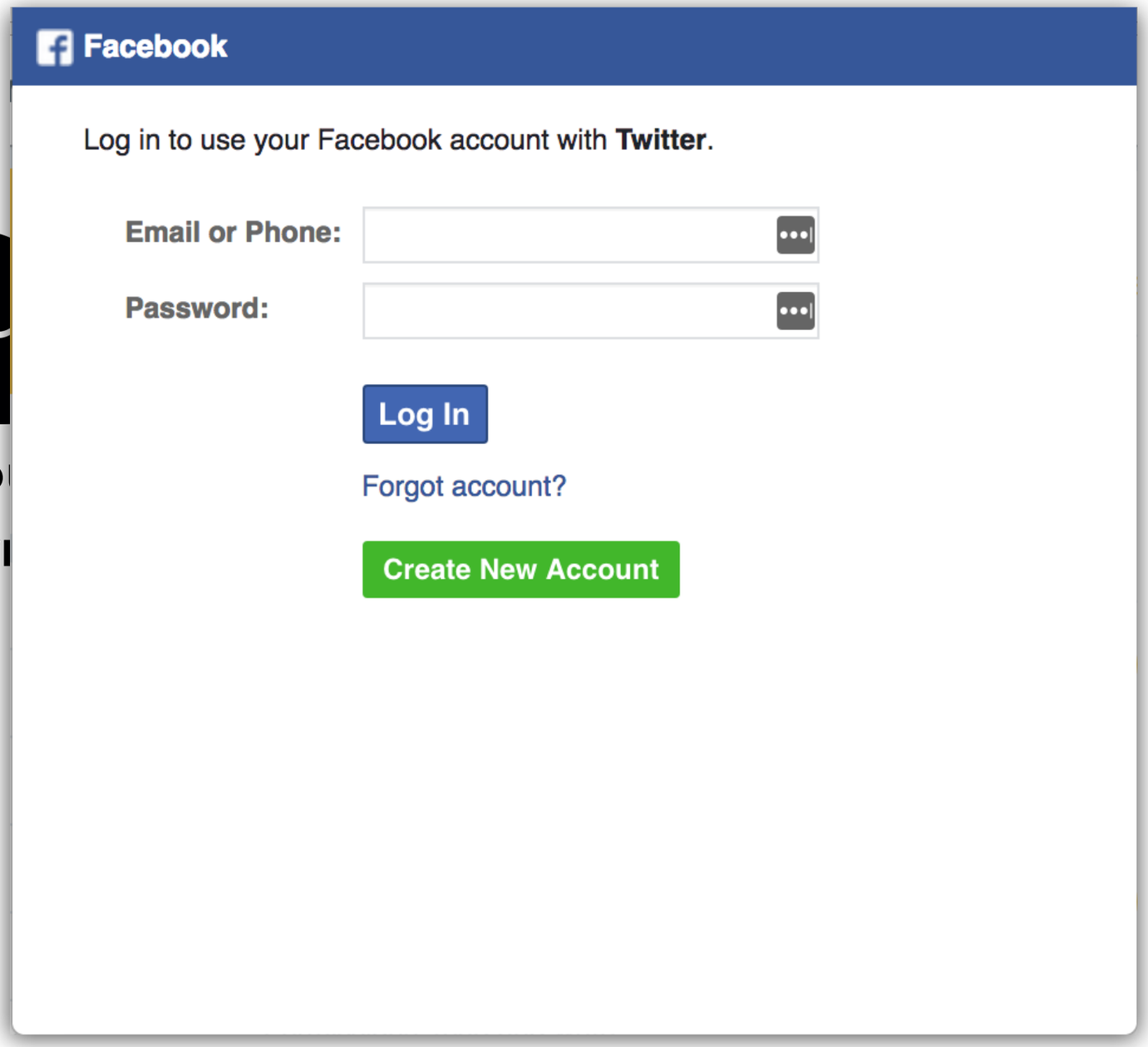
# OAuth: an example delegated authorization flow




The client redirects the user browser to the Facebook Authorization Server to get consent from the user

# OAuth: an example delegated authorization flow

**Authorization Server**



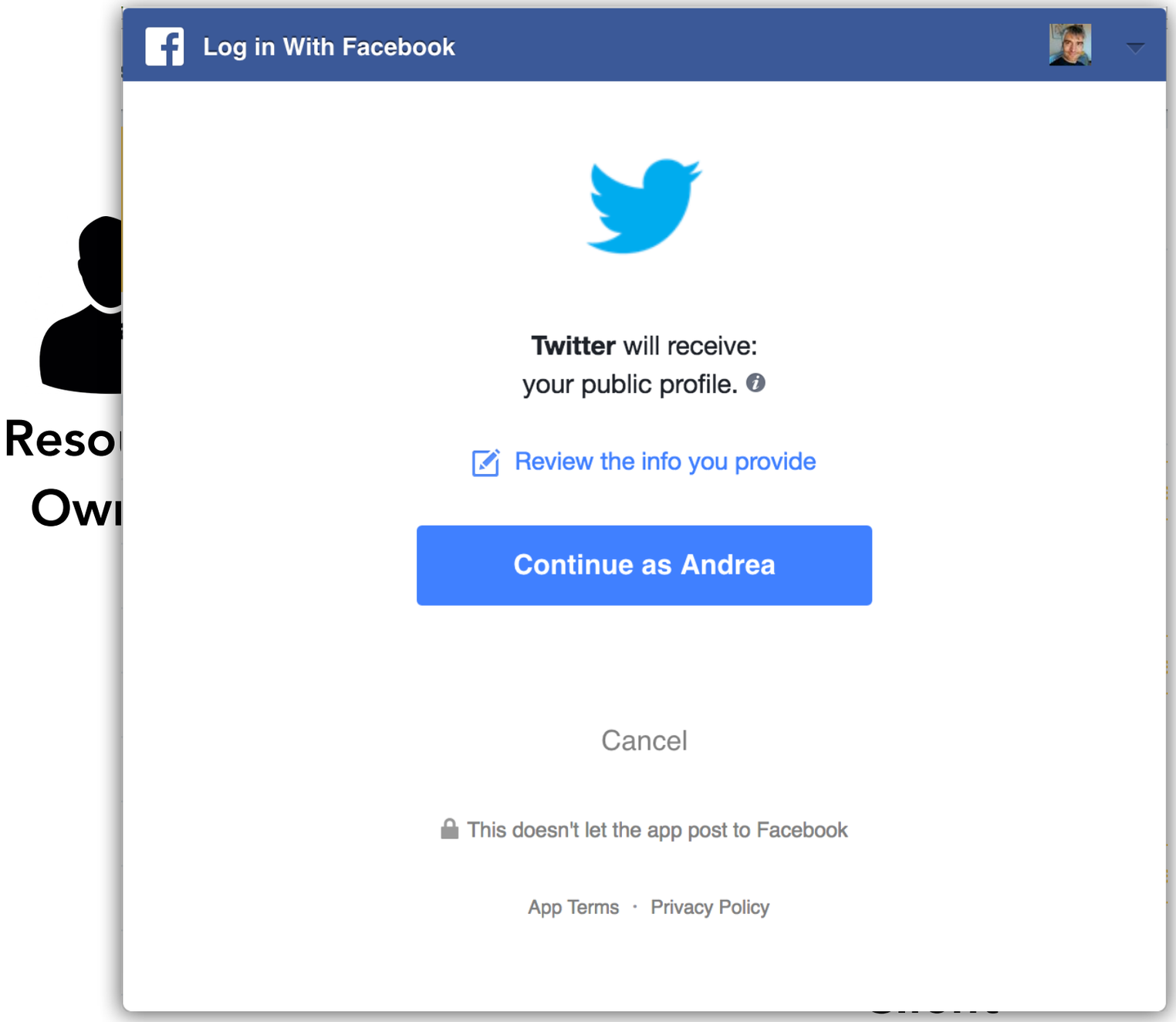

**Reso**

**Owi**

**Client**

The user does not have an active login session @ Facebook, so the Facebook authorization server asks for authentication

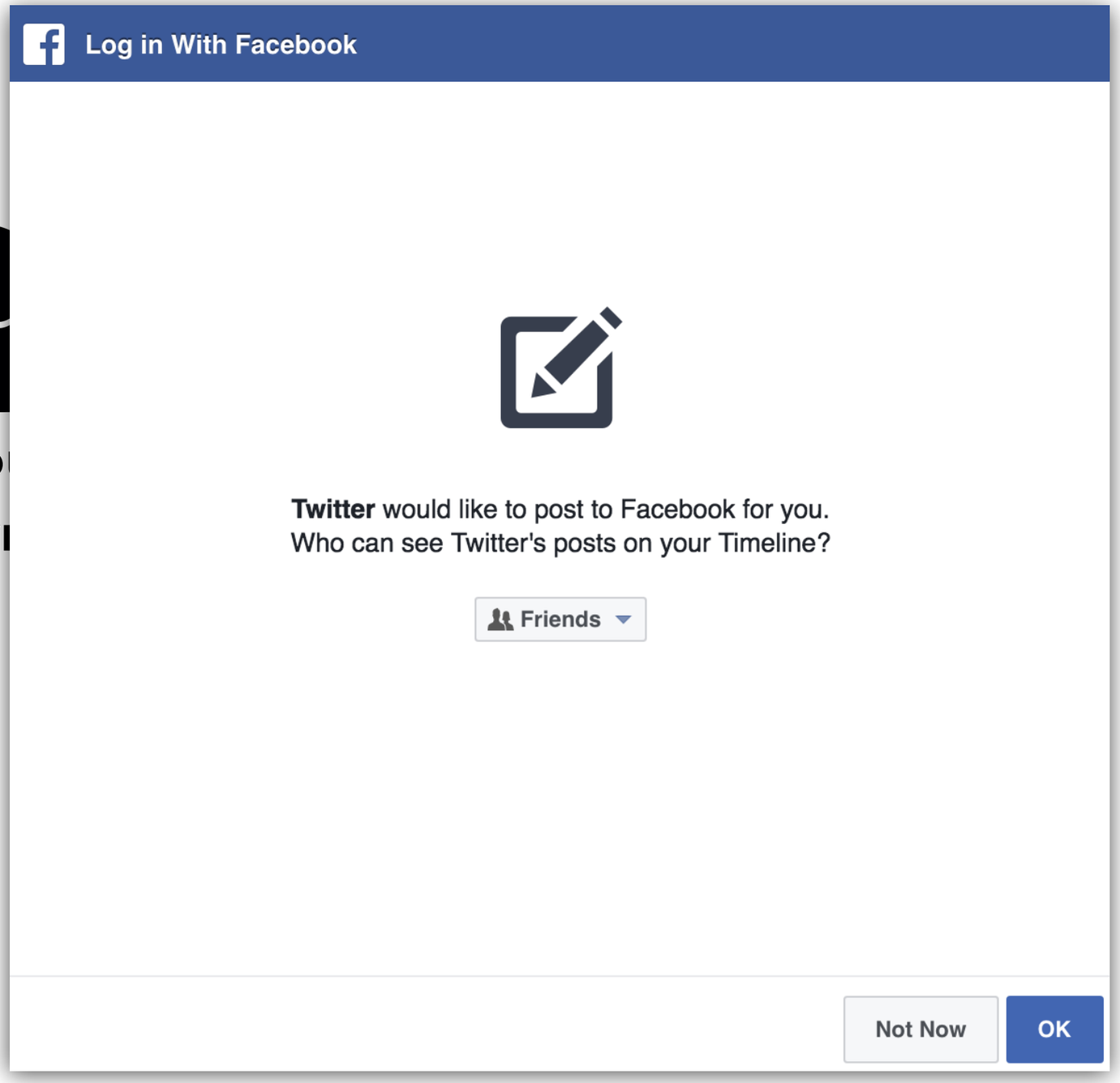
# OAuth: an example delegated authorization flow




Once the user is authenticated, Facebook informs the user that some of his profile information will be shared with Twitter. In order to proceed the user has to approve, i.e. to give his **consent**.

# OAuth: an example delegated authorization flow

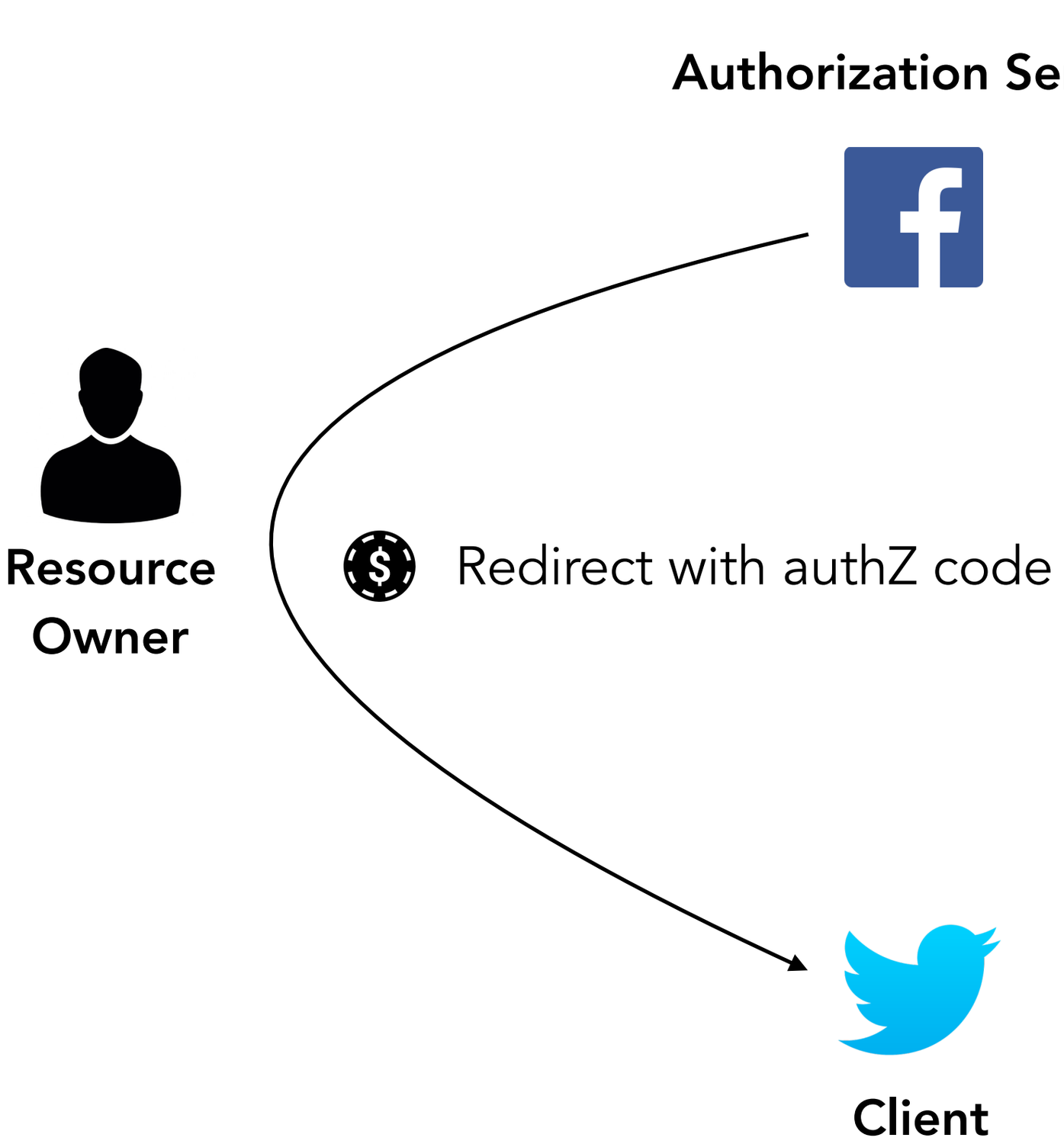
**Authorization Server**




**Reso**
**Owi**

Twitter also requested the ability to write on the user Facebook page, so Facebook asks for consent also for this.

# OAuth: an example delegated authorization flow

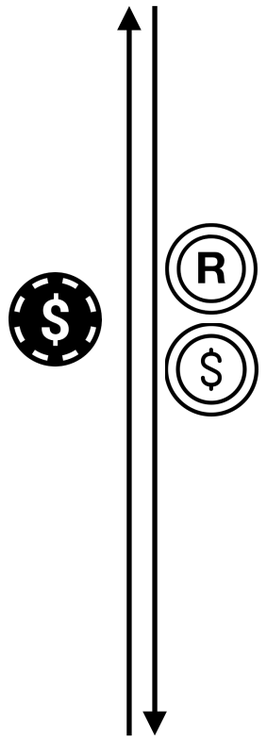



When consent is obtained from the resource owner, Facebook redirects the user to Twitter sending an **authorization code** as a parameter of the HTTP redirect

# OAuth: an example delegated authorization flow

**Authorization Server**

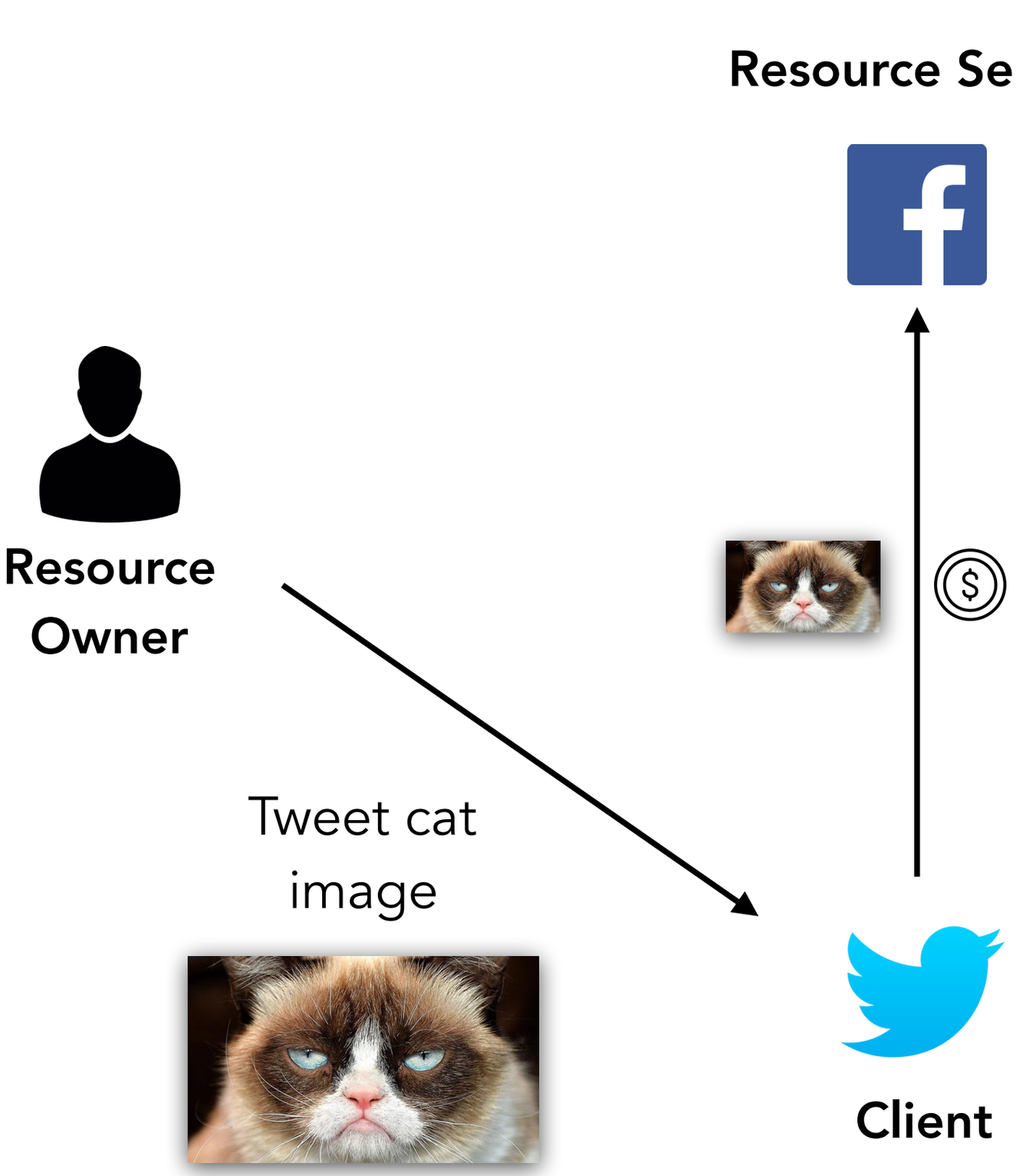
**Resource Owner**



**Client**

Twitter then exchanges the **authorization code** with a short-lived **access token** (and a long-lived **refresh token**). This is a direct message exchange between Twitter and the Facebook AS

# OAuth: an example delegated authorization flow




When the user tweets something, Twitter creates a post on the user timeline via the Facebook APIs (Resource Server). The **access token** is included in API requests for **authorization** purposes

# OAuth: an example delegated authorization flow

**Resource Server**

**Resource Owner**

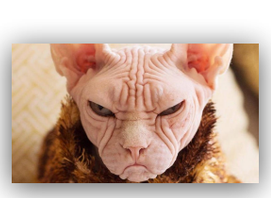
The Facebook APIs (Resource Server) checks the validity of the token and the linked scopes to authorize the creation of a post on the user timeline

**Client**

# OAuth: an example delegated authorization flow

**Resource Server**

**Resource Owner**



Tweet another cat image

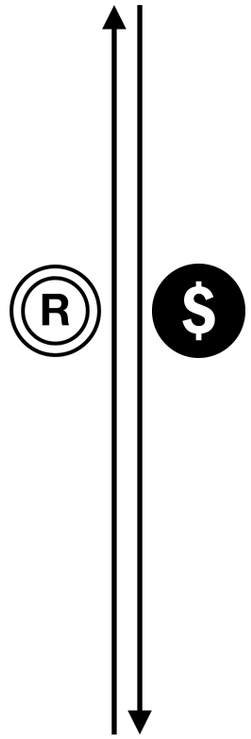
**Client**

After some time the user tweets again, Twitter tries to post again on Facebook, but this time the posting fails since the **access token** has **expired**

# OAuth: an example delegated authorization flow

**Authorization Server**

**Resource Owner**




**Client**

Twitter then presents a **refresh token** to the Facebook AS to obtain a new access token

# OAuth: an example delegated authorization flow

**Resource Server**

**Resource Owner**



And tries to post again with the new token…

**Client**

# OAuth: an example delegated authorization flow

**Resource Server**

**Resource Owner**

And this time the posting succeeds

**Client**

# OpenID Connect: an identity layer for OAuth

OAuth is a **delegated authorization** protocol

- an **access token** states the **authorization rights** of the client application presenting the token to access some resources

OpenID Connect extends OAuth to provide a standard **identity layer**

- i.e. information about **who the user is** and **how it was authenticated** via an additional **ID token (JWT)** and a dedicated **user information query endpoint** at the OpenID Connect Identity provider
- provides ability to establish **login sessions** (SSO)

+

# JSON Web Tokens (JWT)

**JSON Web Token** (JWT) is an open standard that defines a compact, self-contained way of securely transmitting information between parties as a JSON object

JWTs are typically **signed** and, if confidentiality is a requirement, can be **encrypted**.

Header

```
{
  "kid": "rsa1",
  "alg": "RS256"
}
```

.

Body

```
{
  "sub": "e1eb758b-b73c-4761-bfff-adc793da409c",
  "iss": "https://iam-test.indigo-datacloud.eu/",
  "exp": 1482163788,
  "iat": 1482160188,
  "jti": "e7bcb54c-8f67-4a77-8415-37adeb4b958c"
}
```

.

Signature

```
QbOfPrha9kp4e7TknXe88
d8v_9e7V2v2xMAKX10xY4
M3P1wragAhQmyoVQwq-uk
```

# Why OAuth, OpenID Connect and JWT?

Standard, widely adopted in industry

- Do not reinvent the wheel, reuse existing knowledge and tools, extend when needed

Reduced integration complexity at relying services

- Off-the-shelf libraries and components

Authentication-mechanism agnostic

- The AAI is not bound to a specific authentication mechanism

Distributed verification of access and identity tokens

- It scales

# Back to our token-based AAI...

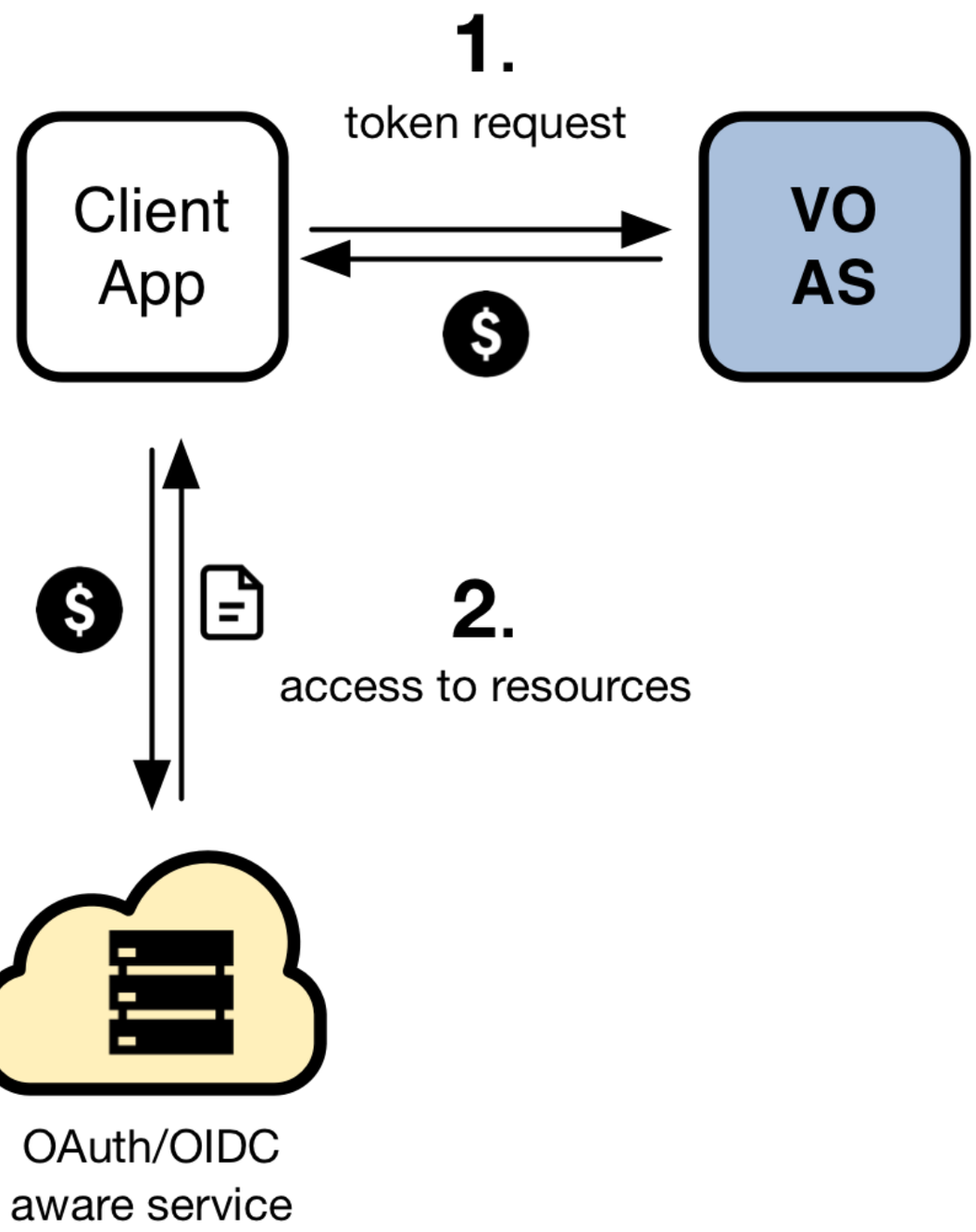
# OAuth and OpenID Connect for WLCG

In order to access resources/services, a **client application** needs an **access token**

The token is obtained from **a VO** (which acts as an OAuth Authorization Server) using standard **OAuth/OpenID Connect** flows

**Authorization** is then **performed at the services** leveraging info extracted from the token:

- **Identity attributes**: e.g., groups
- **OAuth scopes**: authZ labels that are linked to access tokens at token creation time

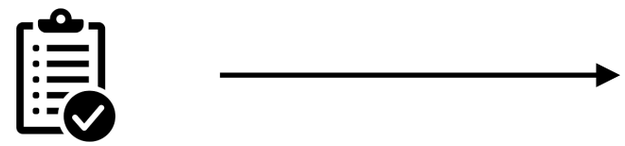


# Attribute-based vs Scope-based Authorization

**Attribute-based authorization:** the token brings information about attribute ownership (e.g., groups/role membership), the service maps these attributes to a local authorization policy

token claims

```
{
  “iss”: “https://cms.wlcg.example”,
  …
  “groups”: “/cms”
}
```

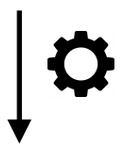



**Scope-based authorization:** the token brings information about which actions should be authorized at a service, the service needs to understand these capabilities and honor them. The authorization policy is managed at the VO level

policy

token claims

```
{
  “iss”: “https://cms.wlcg.example”,
  …
  “scope”: “read:/ upload:/store”
}
```



# INDIGO Identity and Access Management service

**Flexible authentication** support

- (SAML, X.509, OpenID Connect, username/password, …)

**Account linking**

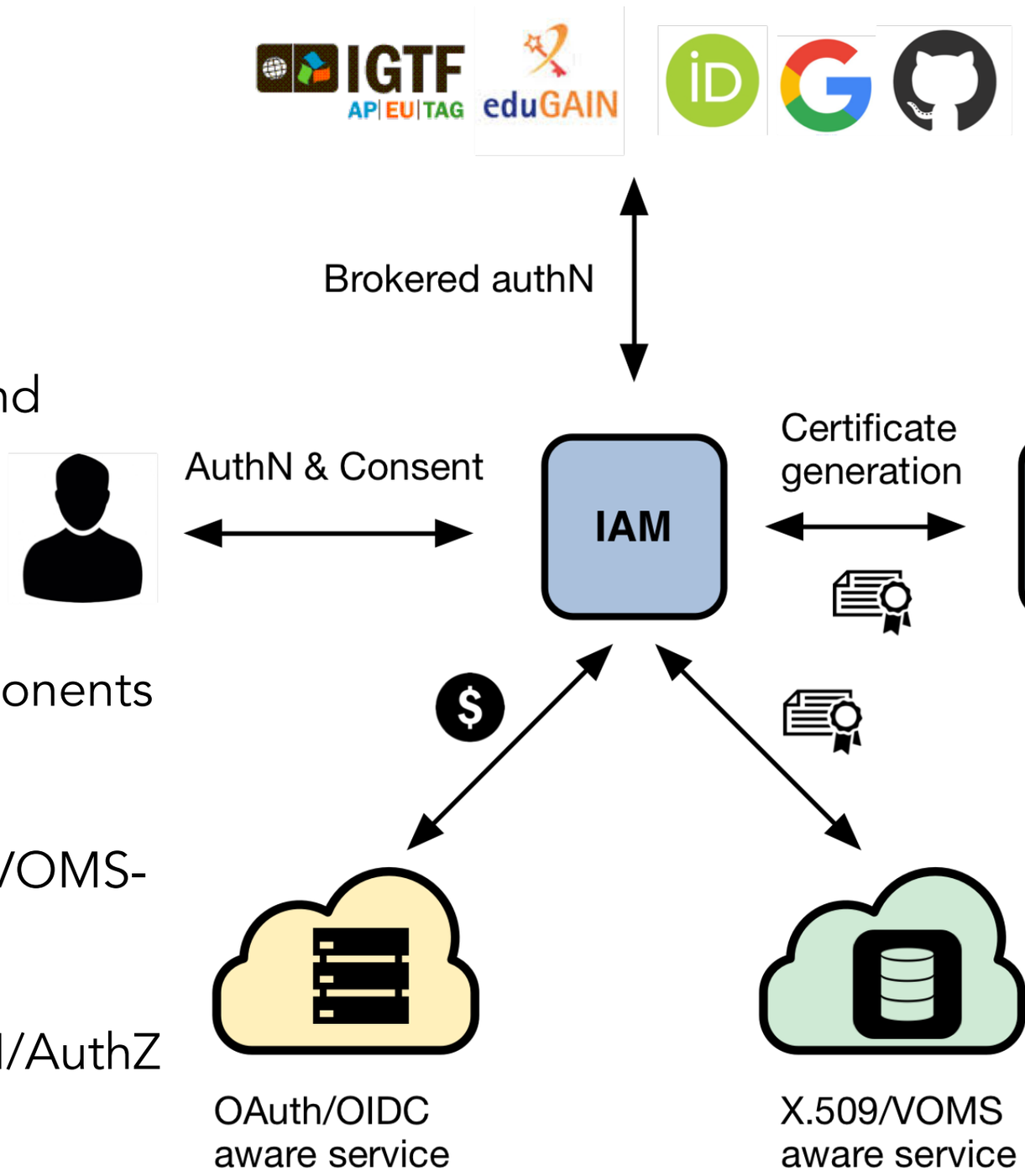
**Registration service** for moderated and automatic user enrollment

**Enforcement of AUP acceptance**

**Easy integration** in off-the-shelf components thanks to **OpenID Connect/OAuth**

**VOMS support,** to integrate existing VOMS-aware services

**Self-contained**, comprehensive AuthN/AuthZ solution



# INDIGO Identity and Access Management service

**Flexible authentication** support

- (SAML, X.509, OpenID Connect, username/password, …)

**Account linking**

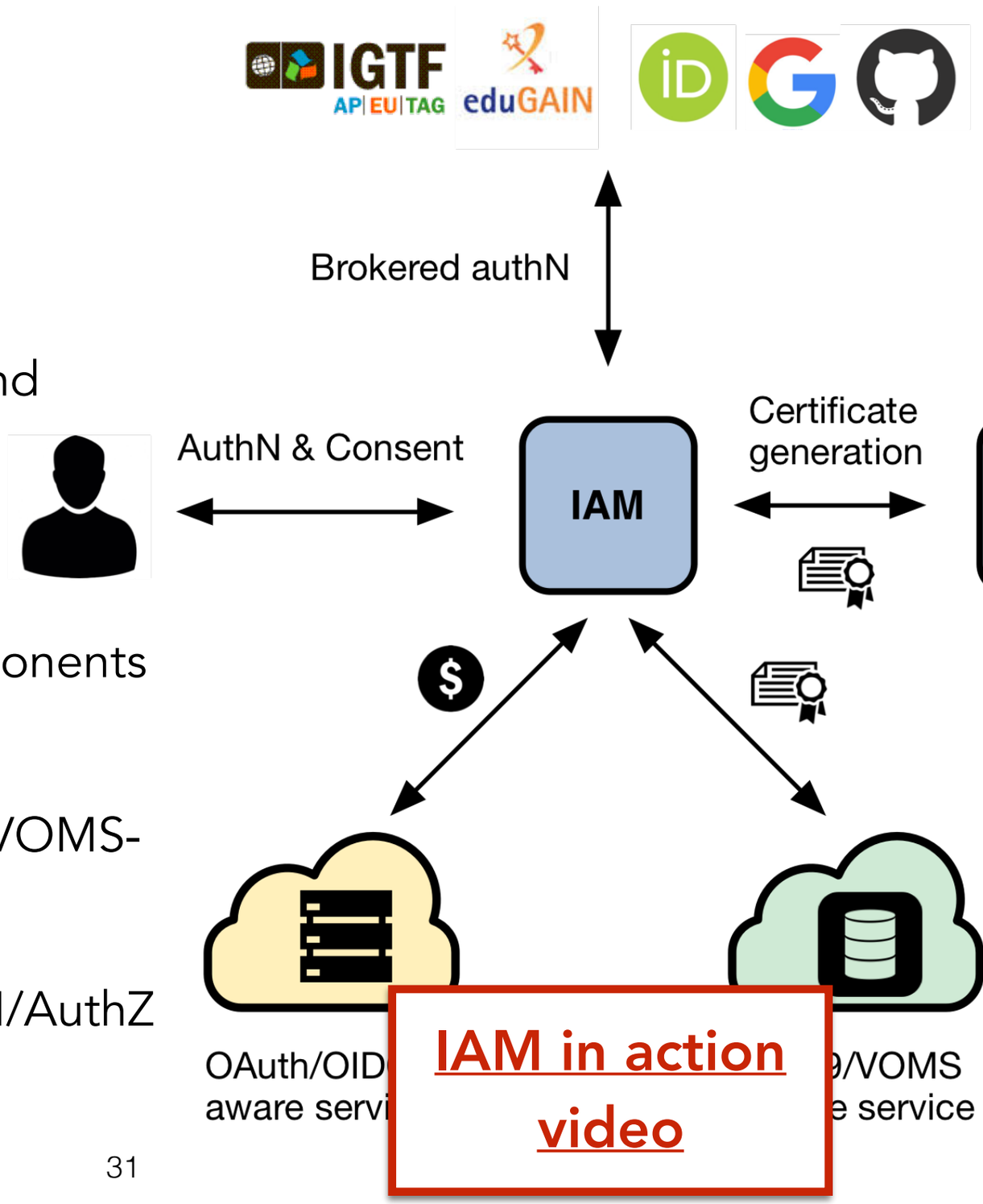
**Registration service** for moderated and automatic user enrollment

**Enforcement of AUP acceptance**

**Easy integration** in off-the-shelf components thanks to **OpenID Connect/OAuth**

**VOMS support,** to integrate existing VOMS-aware services

**Self-contained**, comprehensive AuthN/AuthZ solution

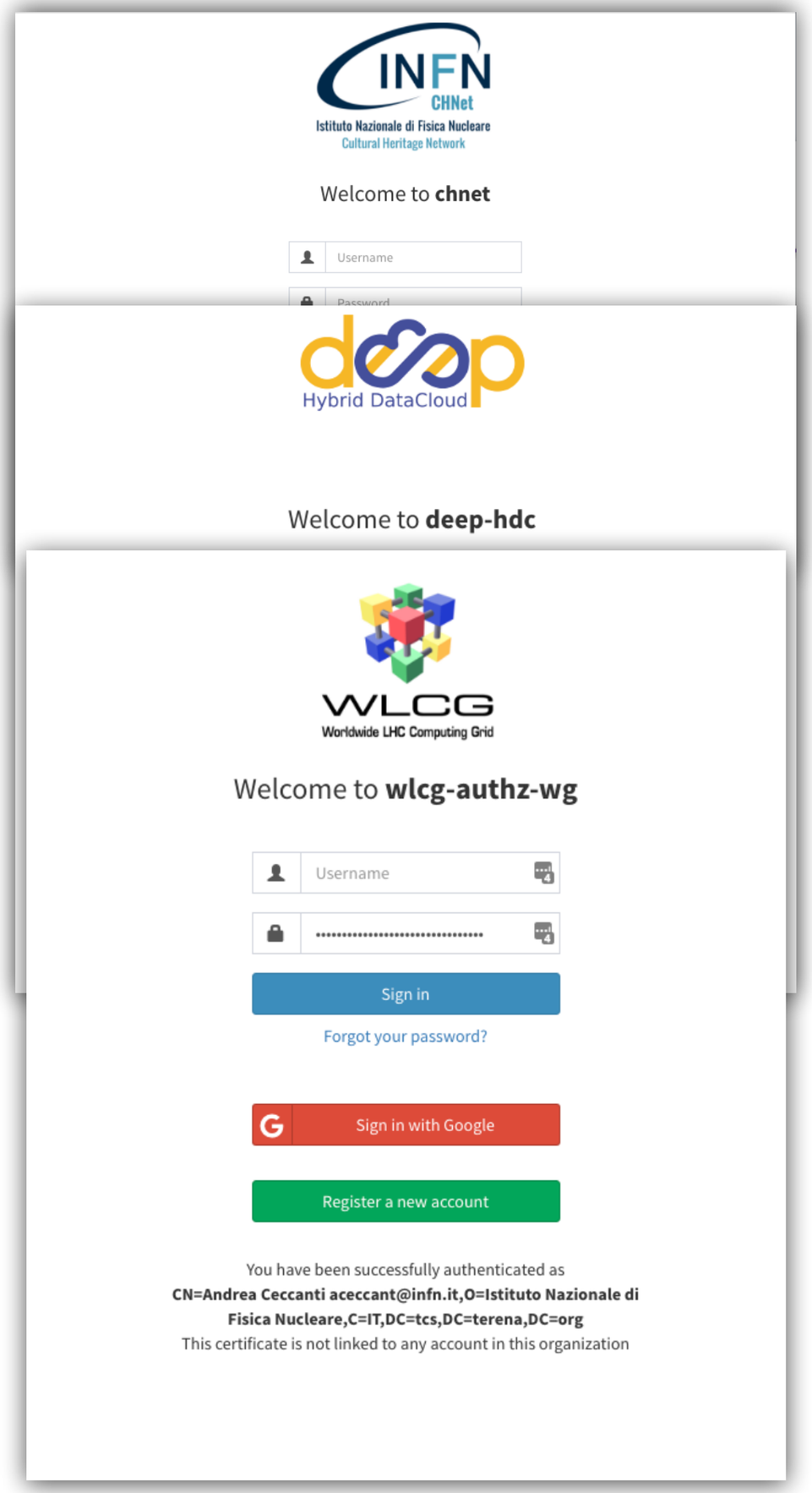


# IAM deployment model

An IAM instance is deployed for a **community** of users sharing resources, the good old **Virtual Organization (VO)** concept

Client applications and services are integrated with this instance via **standard OAuth/OpenID Connect**

The IAM Web appearance can be **customized** to include a **community logo**, **AUP** and **privacy policy** document



# Easy integration with services

**Standard OAuth/OpenID Connect** enable **easy integration** with off-the-shelf services and libraries.

We have successfully integrated IAM with minimal effort with:

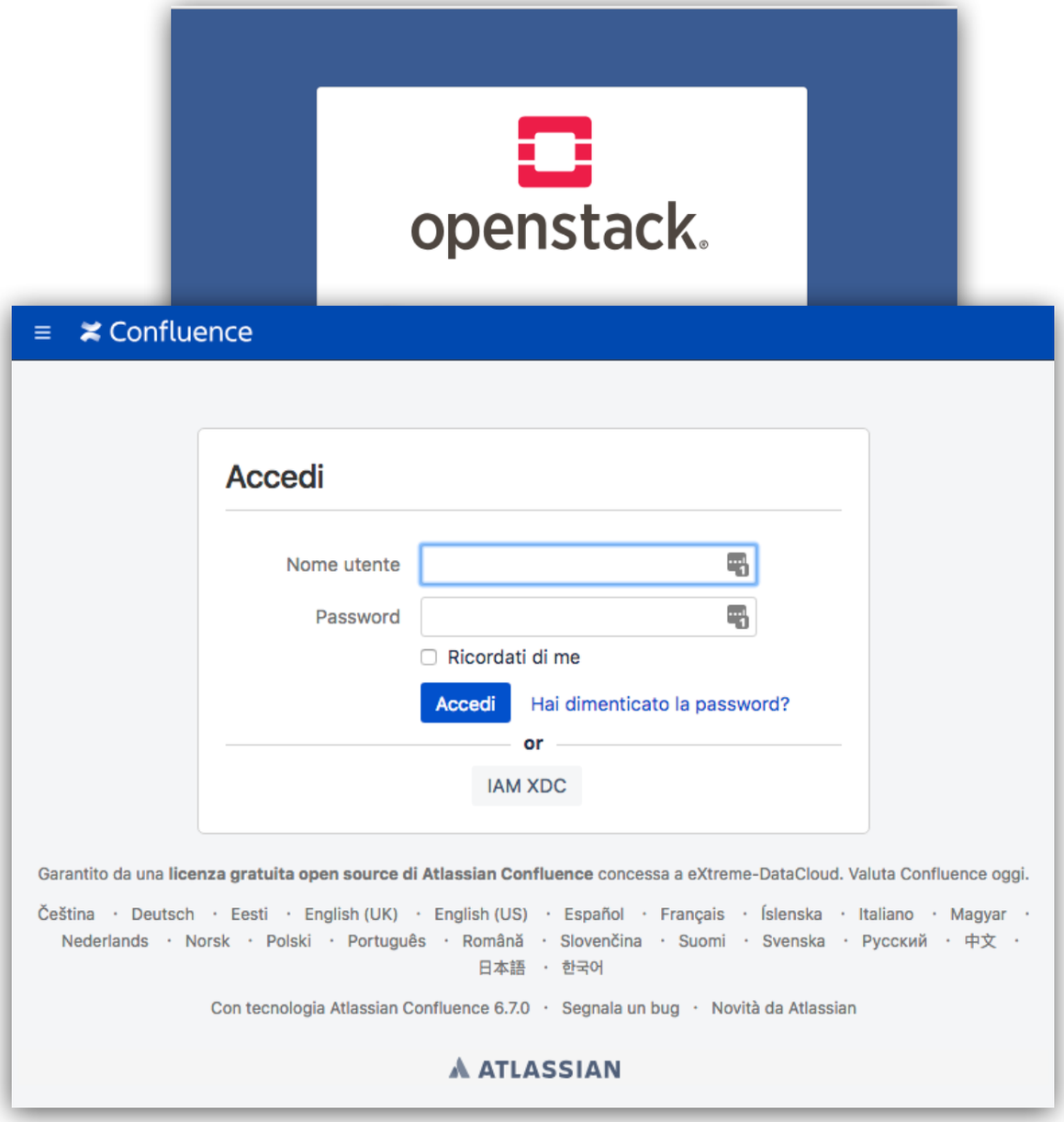
- Openstack
- Atlassian JIRA & Confluence
- Kubernetes
- Moodle
- Rocketchat
- Grafana
- JupyterHub

# Easy integration with services

**Standard OAuth/OpenID Connect** enable **easy integration** with off-the-shelf services and libraries.

We have successfully integrated IAM with minimal effort with:

- Openstack
- Atlassian JIRA & Confluence
- Kubernetes
- Moodle
- Rocketchat
- Grafana
- JupyterHub

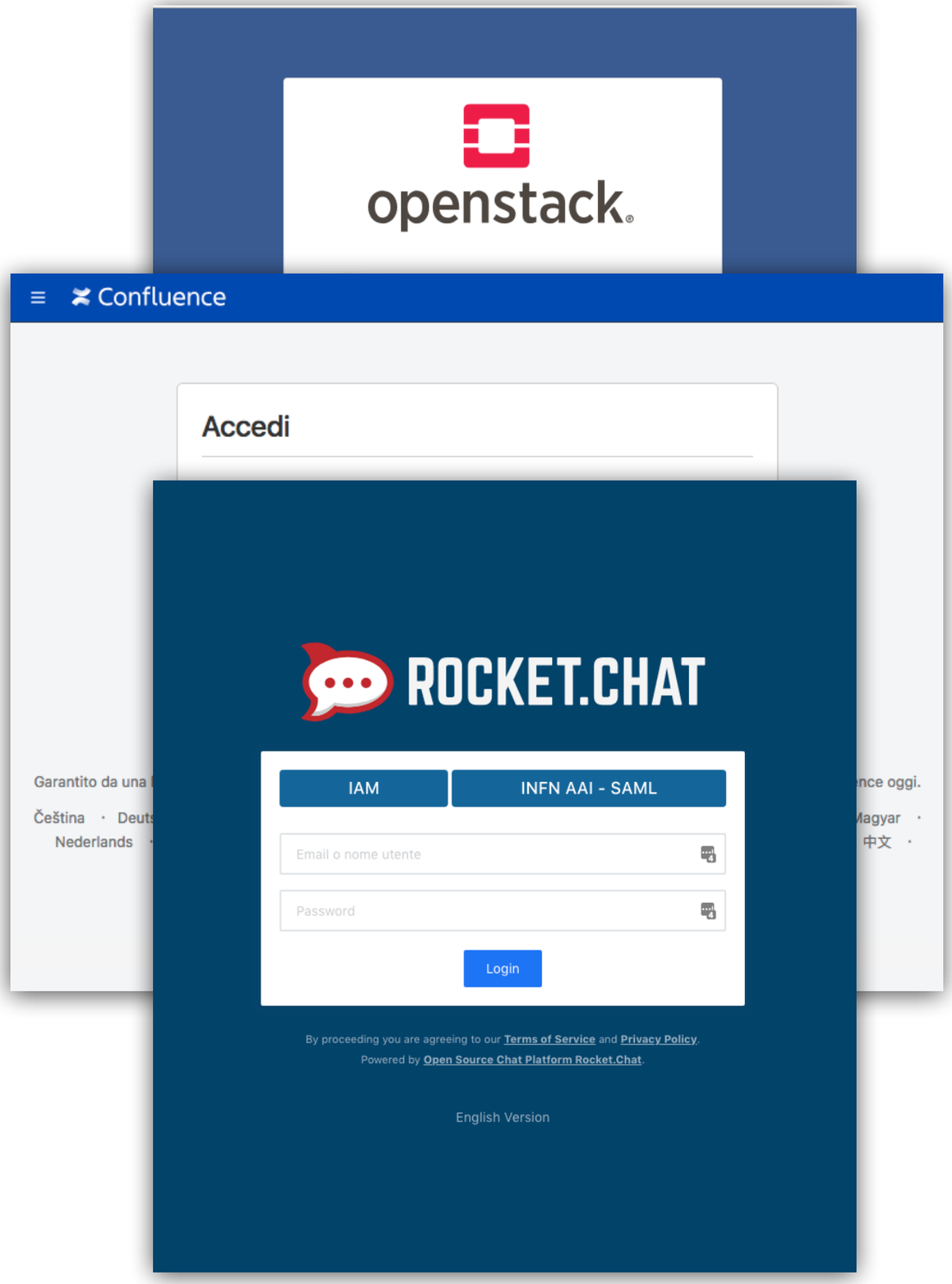


# Easy integration with services

**Standard OAuth/OpenID Connect** enable **easy integration** with off-the-shelf services and libraries.

We have successfully integrated IAM with minimal effort with:

- Openstack
- Atlassian JIRA & Confluence
- Kubernetes
- Moodle
- Rocketchat
- Grafana
- JupyterHub

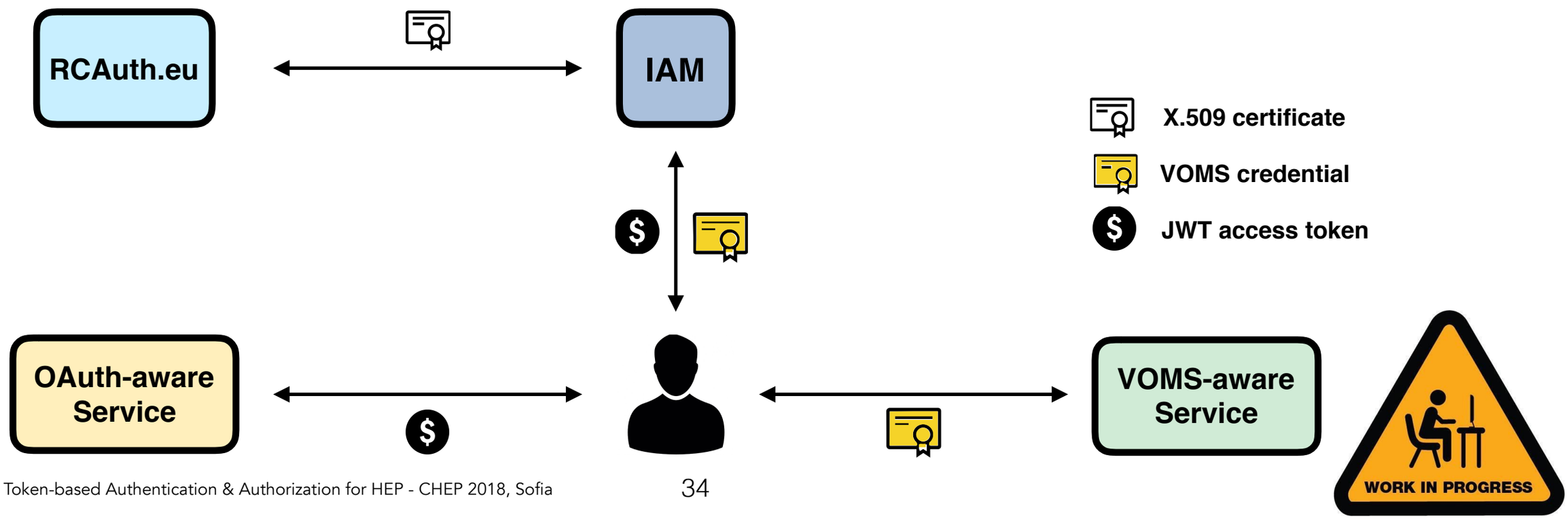


# Seamless transition from an X.509-based AAI

An IAM **VOMS endpoint** exposes authentication and authorization information for an IAM user in the form of a **VOMS attribute certificate, compatible** with existing VOMS clients

Integration with the **RCAuth.eu** online CA allows to generate X.509 certificates on-demand and link them to IAM user memberships

## **Consistent authorization** for VOMS and OAuth/OIDC services



# Related initiatives

**EOSC-Hub AAI**: harmonization across Identity solutions (EGI CheckIn, INDIGO IAM, B2Access, …) for an interoperable EOSC AAI (CHEP 2018 talk)

**SciTokens**: OAuth/JWT profile for capability-based authorization and integration in existing middleware (HTCondor, XRootD, …) (CHEP 2018 talk)

**dCache**: token-based authorization based on macaroons, support for OpenID Connect authentication and initial support for SciTokens (CHEP 2018 talk)

**HTTP LHC Data transfer Ecosystem**: HTTP third-party transfers with token-based authorization (CHEP 2018 talk)

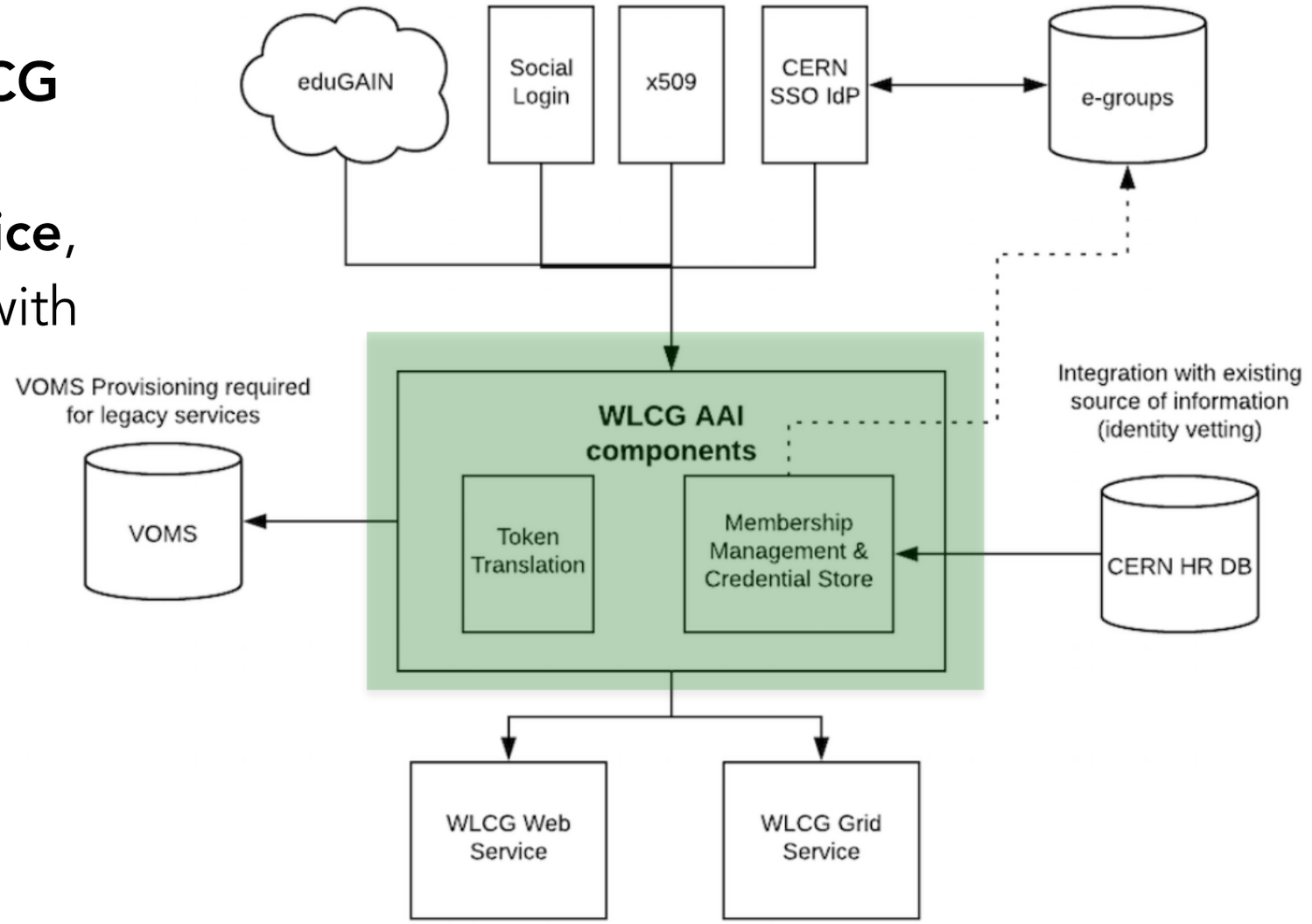
**AARC and FIM4R:** a common AA and policy framework for research communities  and recommendations on how to integrate Federated Identity Management (CHEP 2018 talk)

# The WLCG Authorization WG

https://twiki.cern.ch/twiki/bin/view/LCG/WLCGAuthorizationWG

Main objectives:

- Design and testing of a **WLCG Membership Management and Token Translation service**, facilitated by pilot projects with the support of AARC
- Definition of a **token-based authentication and authorization profile for WLCG**



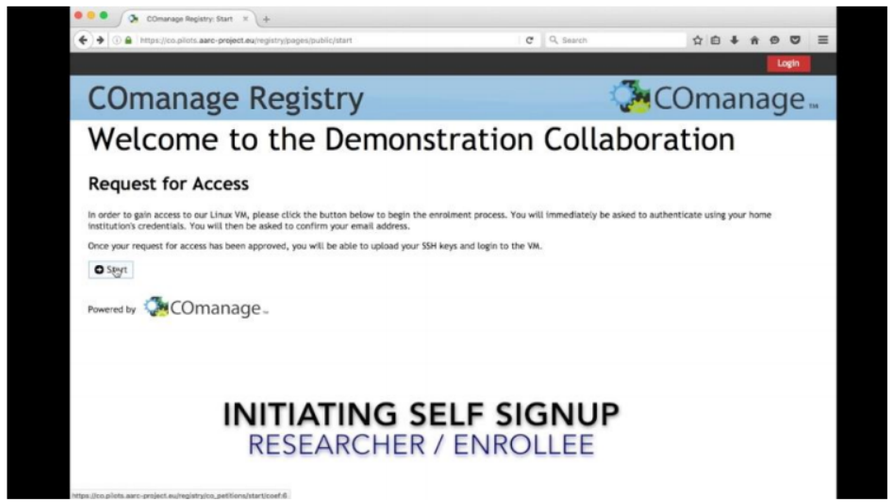
# The WLCG Authorization WG

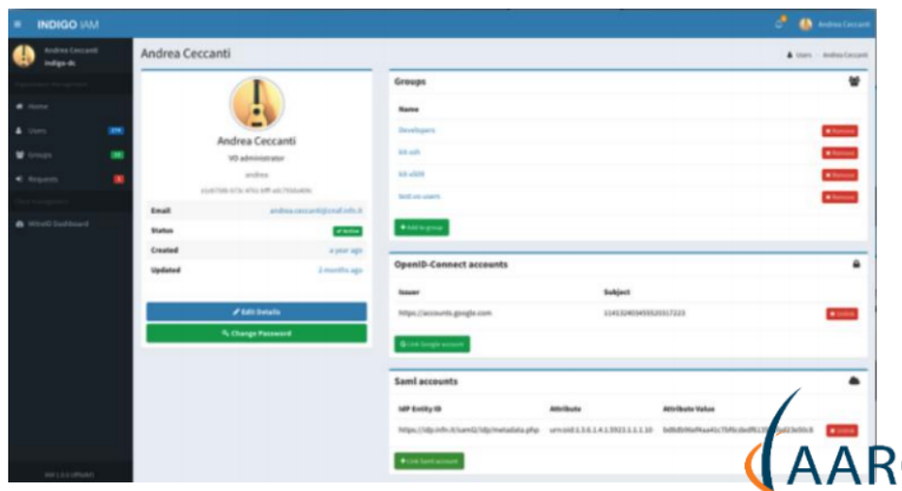
https://twiki.cern.ch/twiki/bin/view/LCG/WLCGAuthorizationWG

M

## AAI Pilot Projects

- Two solutions appear to meet the majority of requirements
  - EGI Check-in & COmanage
  - INDIGO IAM
- Additional integration required for
  - VOMS provisioning & lookup
  - CERN HR DB integration
  - AUP re-signing




h existing
rmation
tting)

R DB

WLCG Worldwide LHC Computing Grid

WLCG AuthZ WG

AARC

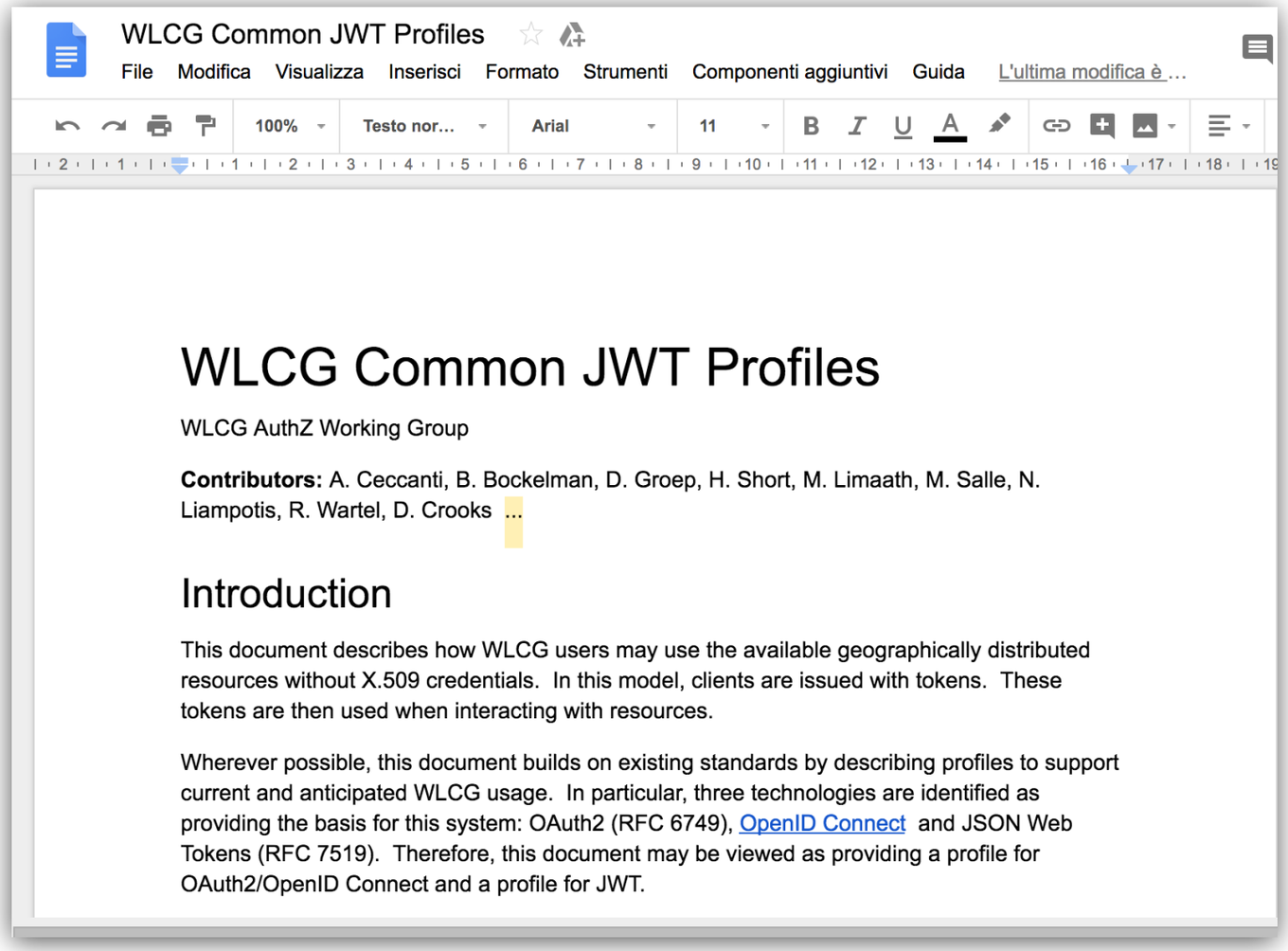
14

# A common profile for Token-based AuthN/AuthZ

How is **authentication** and **authorization** information encoded in **identity** and **access tokens**?

How is **trust** established between parties exchanging tokens?

What's the recommended **token lifetime?**


WLCG Common JWT Profiles

File Modifica Visualizza Inserisci Formato Strumenti Componenti aggiuntivi Guida L'ultima modifica è …

100% Testo nor… Arial 11

WLCG Common JWT Profiles

WLCG AuthZ Working Group

**Contributors:** A. Ceccanti, B. Bockelman, D. Groep, H. Short, M. Limaath, M. Salle, N. Liampotis, R. Wartel, D. Crooks ...

Introduction

This document describes how WLCG users may use the available geographically distributed resources without X.509 credentials. In this model, clients are issued with tokens. These tokens are then used when interacting with resources.

Wherever possible, this document builds on existing standards by describing profiles to support current and anticipated WLCG usage. In particular, three technologies are identified as providing the basis for this system: OAuth2 (RFC 6749), OpenID Connect and JSON Web Tokens (RFC 7519). Therefore, this document may be viewed as providing a profile for OAuth2/OpenID Connect and a profile for JWT.

Approach:

**rely on existing standards as much as possible, extend only when needed**

# Summary

Moving **beyond X.509** certificates and VOMS is recognized as a key challenge for HEP computing to

- improve usability
- simplify the middleware stack thus reducing development and maintenance costs

Convergence across initiatives and research infrastructures on moving towards **standards-based token authentication & authorization**

- based on OAuth, OpenID Connect, JWTs

**INDIGO IAM** represents the **evolution of VOMS** and is one of the solutions under evaluation by the WLCG AuthZ WG that can enable a **smooth transition** between the current and the future token-based WLCG AAI

The **WLCG Authorization WG** is bringing the experts together to define the requirements for this transition, a common profile for token-based authentication and authorization and assess existing solutions

# Thanks for your attention.

# Questions?

# Backup slides

# Useful references

IAM @ GitHub: https://github.com/indigo-iam/iam

IAM documentation: https://indigo-iam.github.io/docs

WLCG Authorization WG: https://twiki.cern.ch/twiki/bin/view/LCG/WLCGAuthorizationWG

IAM in action video: https://www.youtube.com/watch?v=1rZlvJADOnY

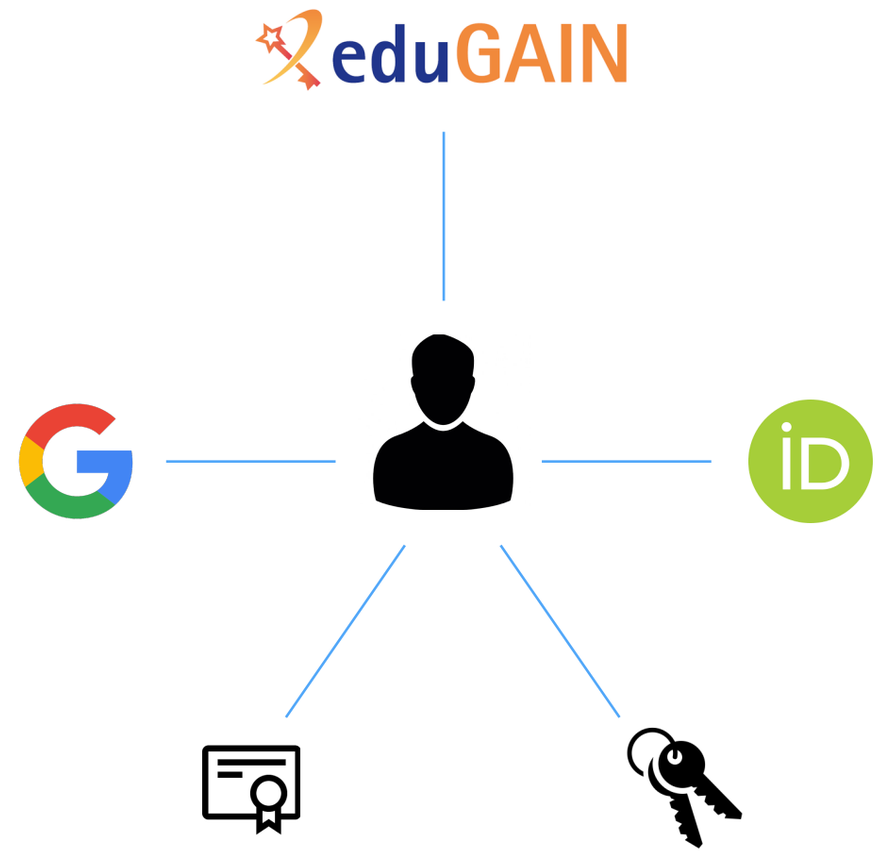
Contacts:

- andrea.ceccanti@cnaf.infn.it
- indigo-aai.slack.com

# Flexible authentication & account linking

Authentication supported via

- **local username/password** credentials (created at registration time)
- **SAML** Home institution IdP (e.g., EduGAIN)
- **OpenID Connect** (Google, Microsoft, Paypal, ORCID)
- **X.509** certificates




Users can link any of the supported authentication credentials to their IAM account at registration time or later

To link an external credential/account, the user has to **prove** that he/she owns such account

# User enrollment & registration service

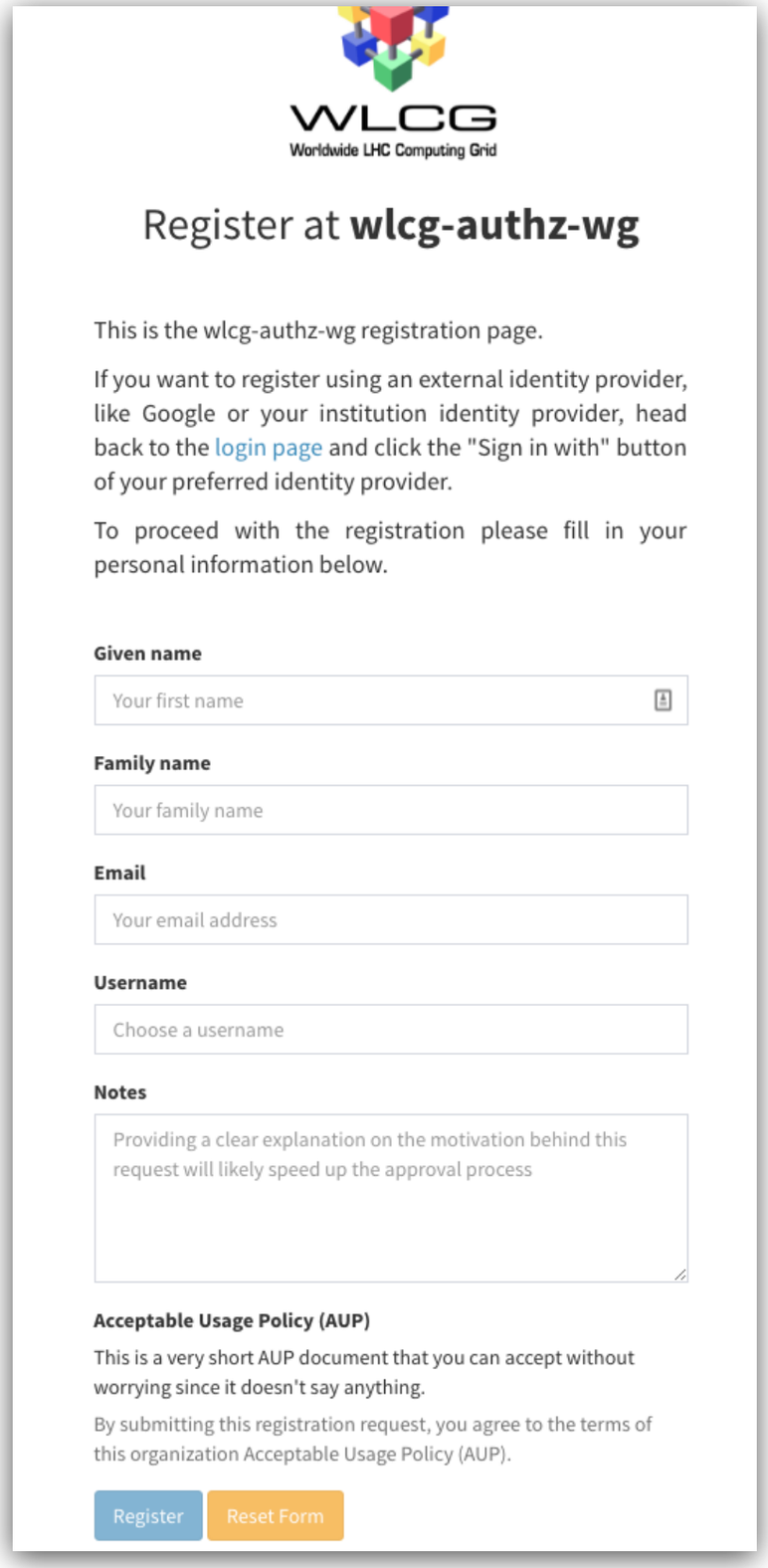
IAM supports two **enrollment flows:**

## **Admin-moderated** flow

- The applicant fills basic registration information, accepts AUP, proves email ownership
- VO administrators are informed by email and can approve or reject  incoming membership requests
- The applicant is informed via email of the administrator decision

## **Automatic-enrollment** flow

- Users authenticated at **trusted**, **configurable** SAML IdPs are automatically on-boarded, without administrator approval



# User enrollment & registration service

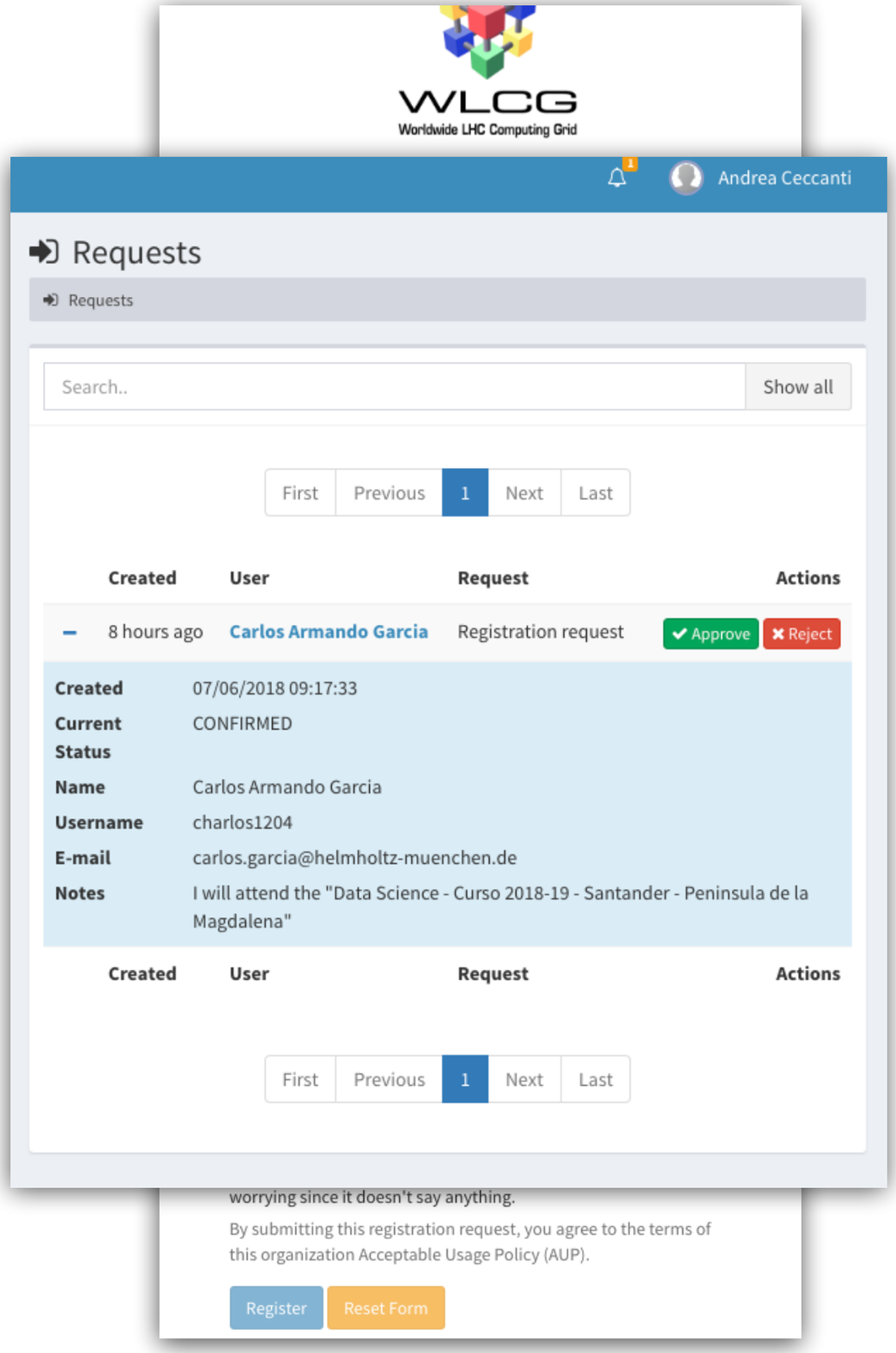
IAM supports two **enrollment flows:**

**Admin-moderated** flow

- The applicant fills basic registration information, accepts AUP, proves email ownership
- VO administrators are informed by email and can approve or reject  incoming membership requests
- The applicant is informed via email of the administrator decision

**Automatic-enrollment** flow

- Users authenticated at **trusted**, **configurable** SAML IdPs are automatically on-boarded, without administrator approval

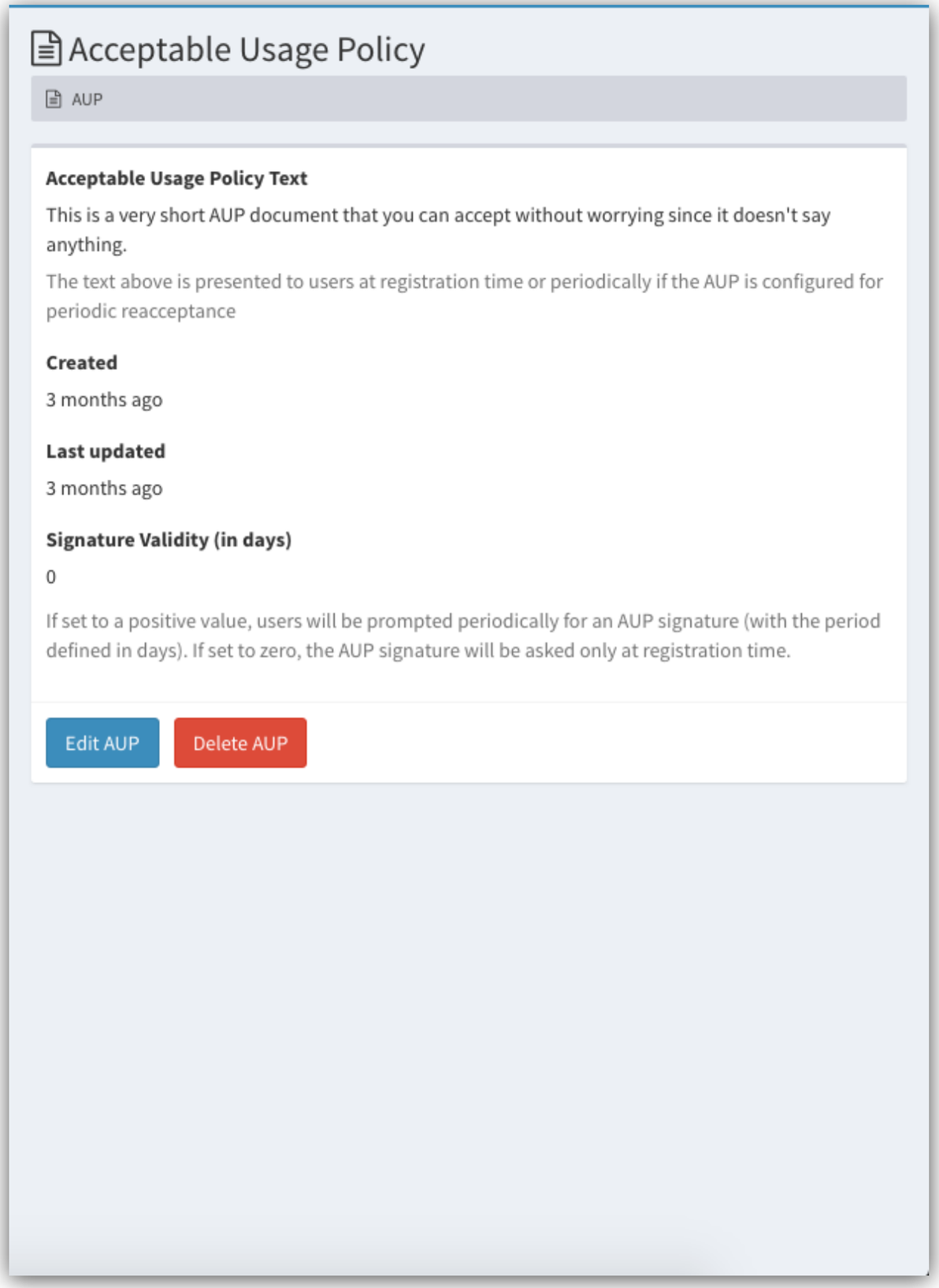


# AUP enforcement support

**AUP acceptance**, if enabled, can be configured to be:

- requested once at user registration time
- periodically, with configurable period

User cannot login to the system (and as such be authenticated at authorized at services) unless the **AUP** has been accepted



# IAM deployment strategies

IAM is a **Spring Boot** application

- currently based on the MitreID Connect
- deployed behind an **NGINX**
- stores data in a **MariaDB**/ **MySQL** database

## Horizontally scalable

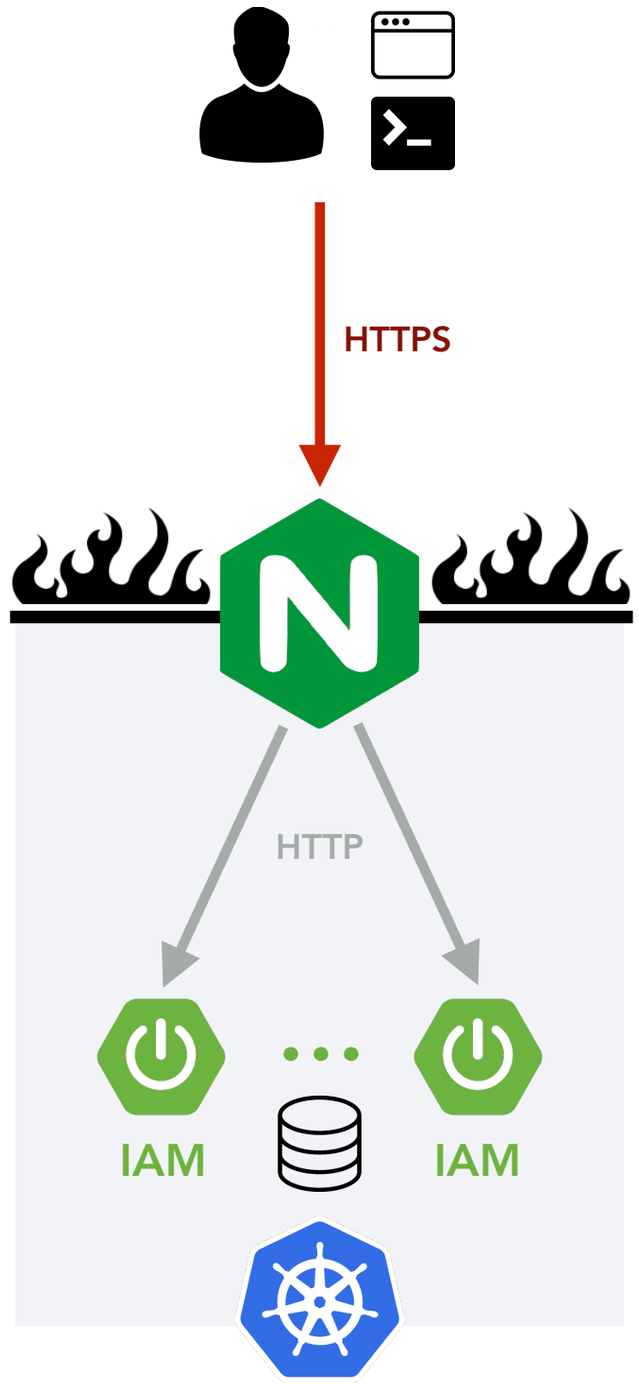
- all state persisted in the database

We deploy IAM as a **containerized** service on top of **Kubernetes**

- autoscaling, zero downtime rolling updates

Packages available for

- CENTOS 7, UBUNTU 1604



# IAM Software Quality

Aim to have **>90% unit test coverage on all code**:

- now 24k LoC, 85.6% branch coverage, >800 tests

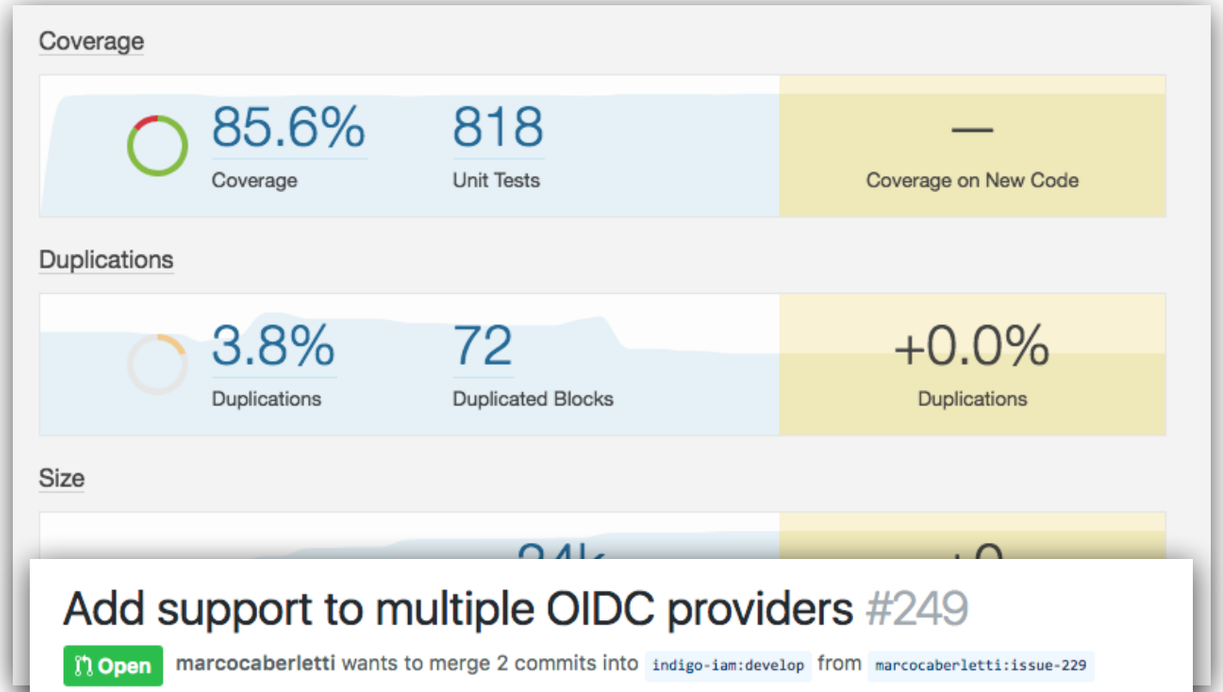
**Open**, **test-driven** development process

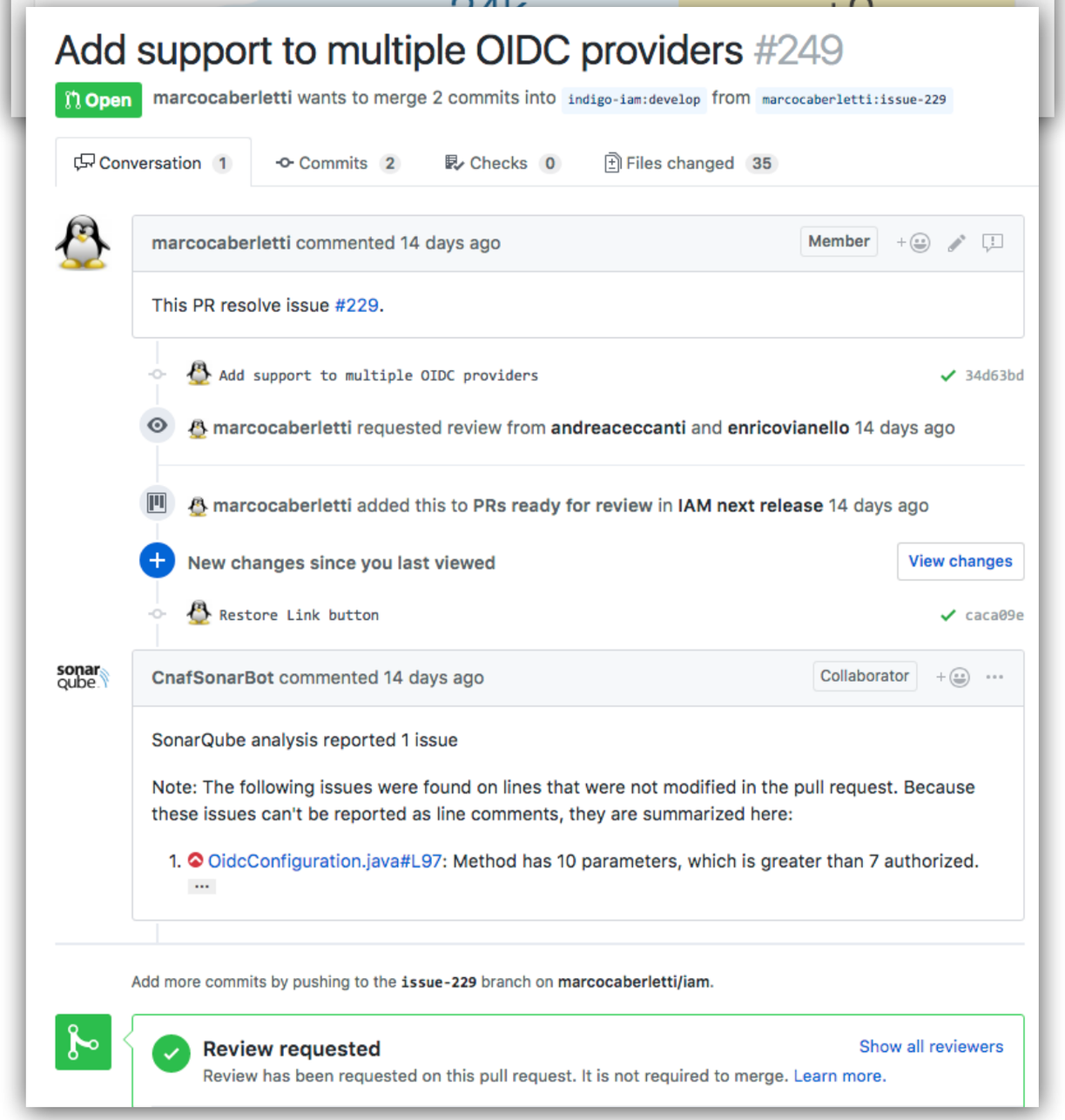
**Static analysis** tools

- SonarCube IAM page

**Multiple test suites**

- **Unit tests**
- **Frontend test suite** (based on Selenium and Robot framework)
- **Deployment tests** (in CI)



# IAM evolution: porting to Keycloak

IAM 2 (in development) will be based on Keycloak

- Powerful RedHat SSO solution
- Vibrant community: > 250 GitHub contributors
- LDAP/Kerberos integration
- Multi-tenancy

We will **focus on what not already provided** by Keycloak

- flexible registration service
- X.509 and VOMS authentication support

**Improved flexibility and sustainability**