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

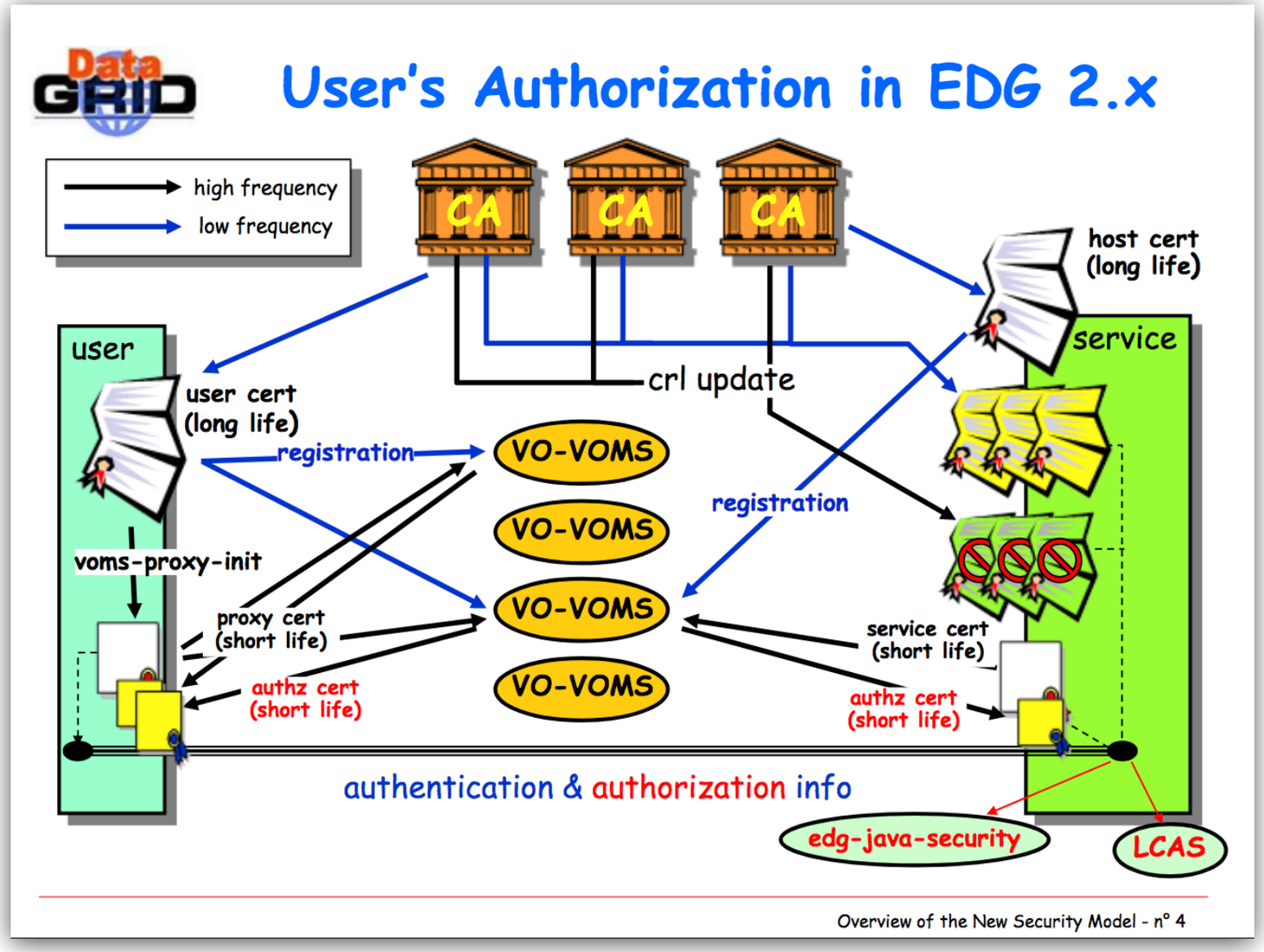
Andrea Ceccanti
INFN CNAF

Atlas Jamboree
CERN, March 8th 2019

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

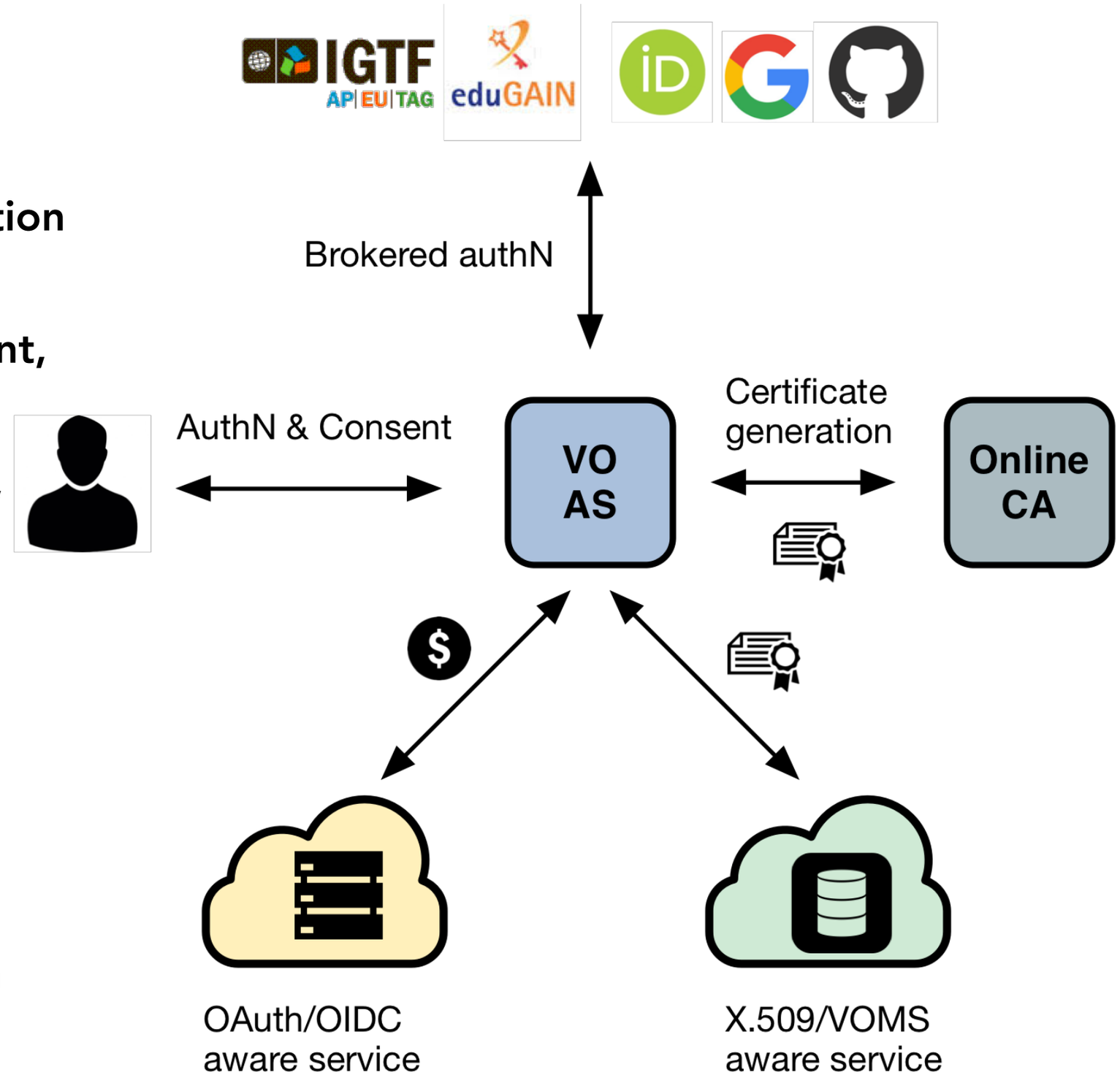
Token translation

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

A token-based AAI for WLCG

Introduce a central VO-scoped service that

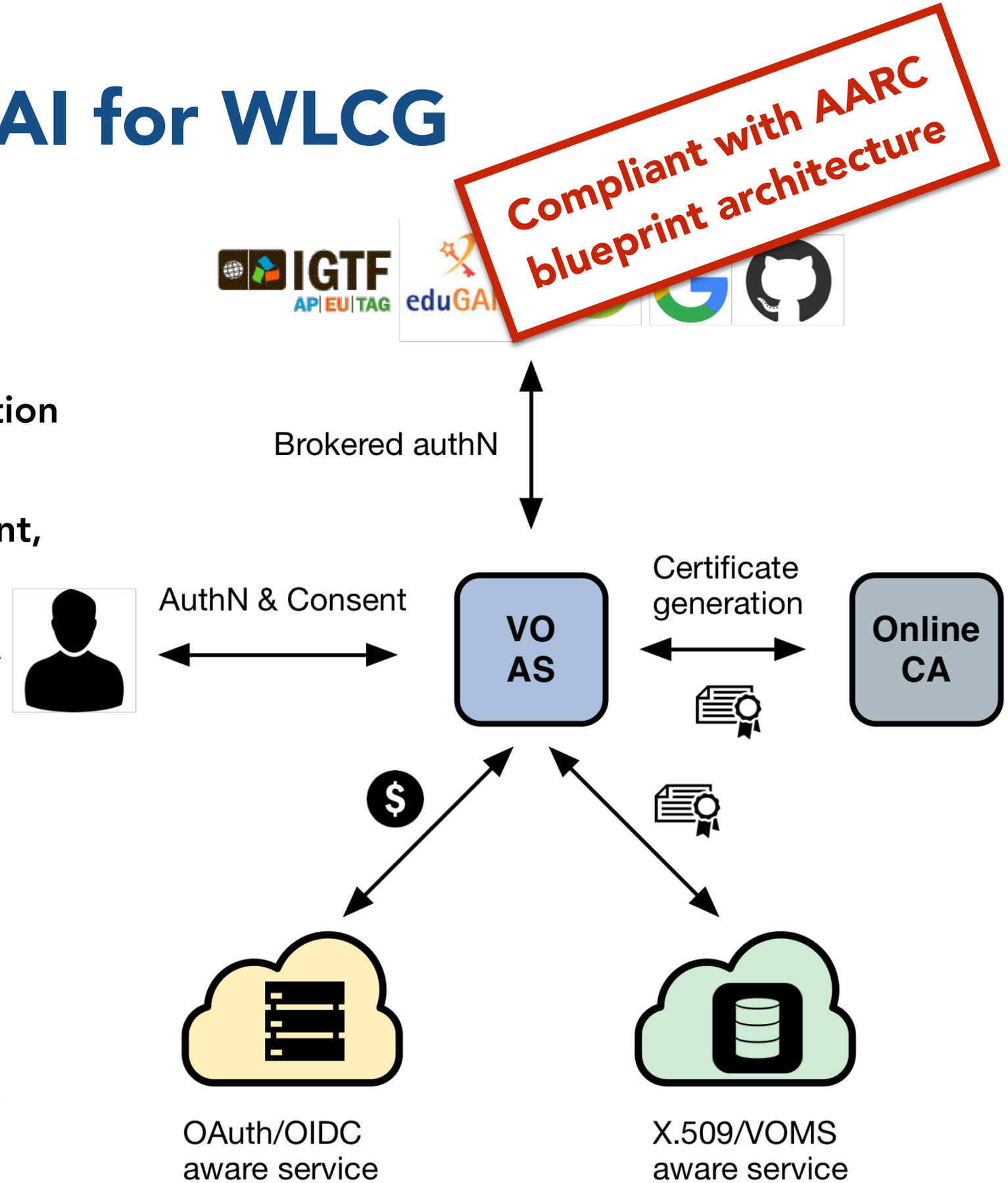
- supports **multiple authentication mechanisms**
- provides users with a **persistent, VO-scoped** identifier
- exposes **identity information**, **attributes** and **capabilities** to services via **JWT** tokens and standard **OAuth & OpenID Connect** protocols
- can integrate existing **VOMS**-aware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


```
{
  "sub": "e1eb758b-b73c-4761-bfff-adc793da409c",
  "aud": "iam-client test",
  "iss": "https://iam-test.indigo-datacloud.eu/",
  "exp": 1507726410,
  "iat": 1507722810,
  "jti": "39636fc0-c392-49f9-9781-07c5eda522e3"
}
```

OAuth: a delegated authorization framework

OAuth defines how **controlled delegation of privileges** can happen among collaborating services

Provides answers to questions like:

- How can an application request access to protected resources?
  - How can I obtain **an access token**?
- How is authorization information exchanged across parties?
  - How is the **access token** presented to **protected resources**? (i.e. API)

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...

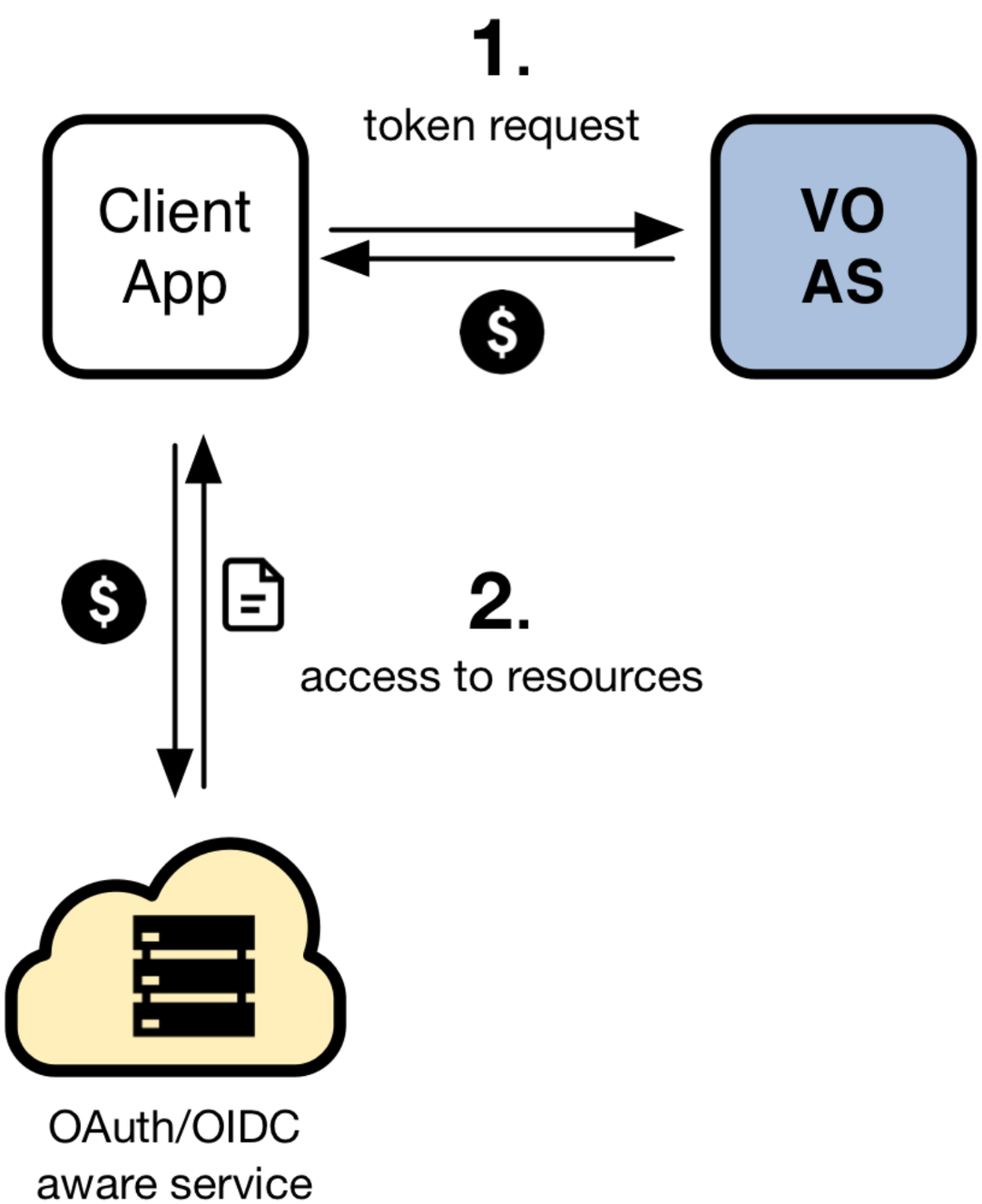
OAuth and OpenID Connect for WLCG

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**, roles, …
- **OAuth scopes**: capabilities linked to access tokens at token creation time

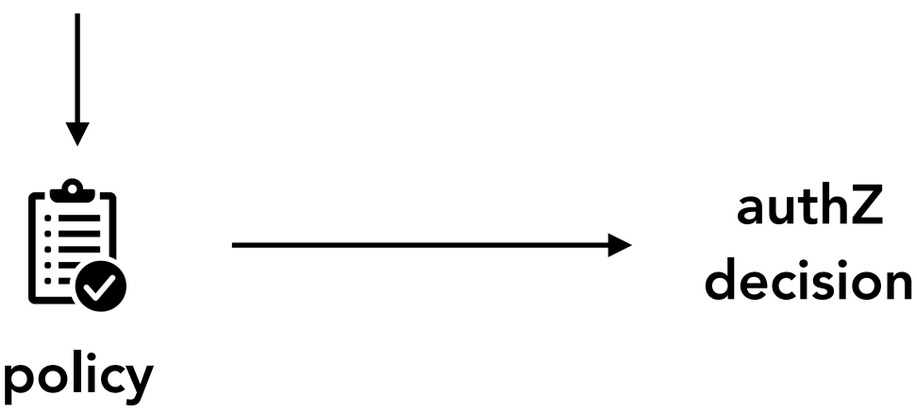


Identity-based vs Scope-based Authorization

Identity-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

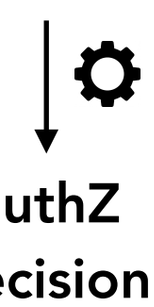
```
{
  "iss": "https://cms.wlcg.example",
  …
  "groups": "/cms"
}
```



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

token claims

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

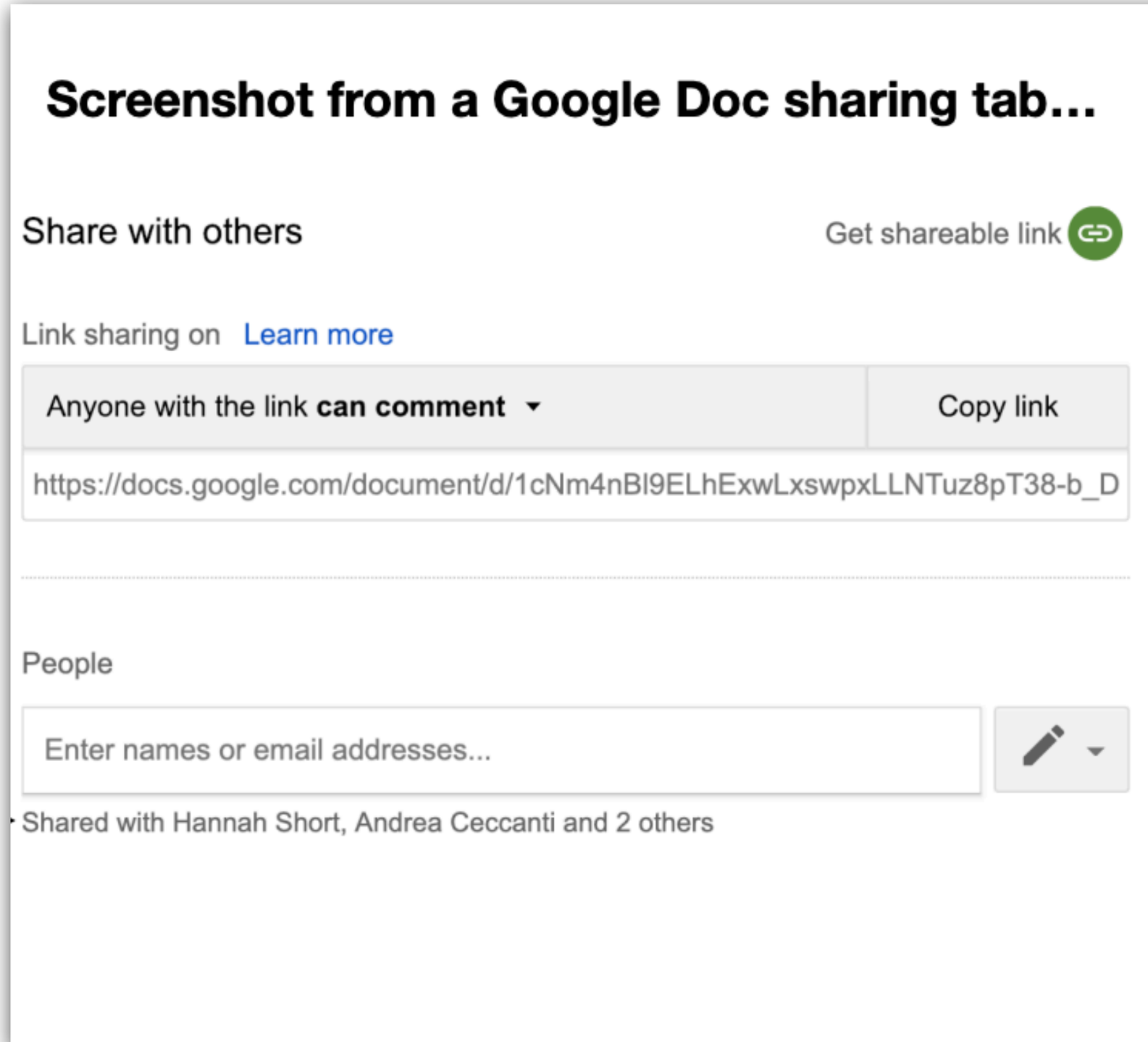


Identity-based vs Scope-based Authorization

The two models can coexist, even in the context of the same application!

scope-based authZ

identity-based authZ



* Slide courtesy of B. Bockelman

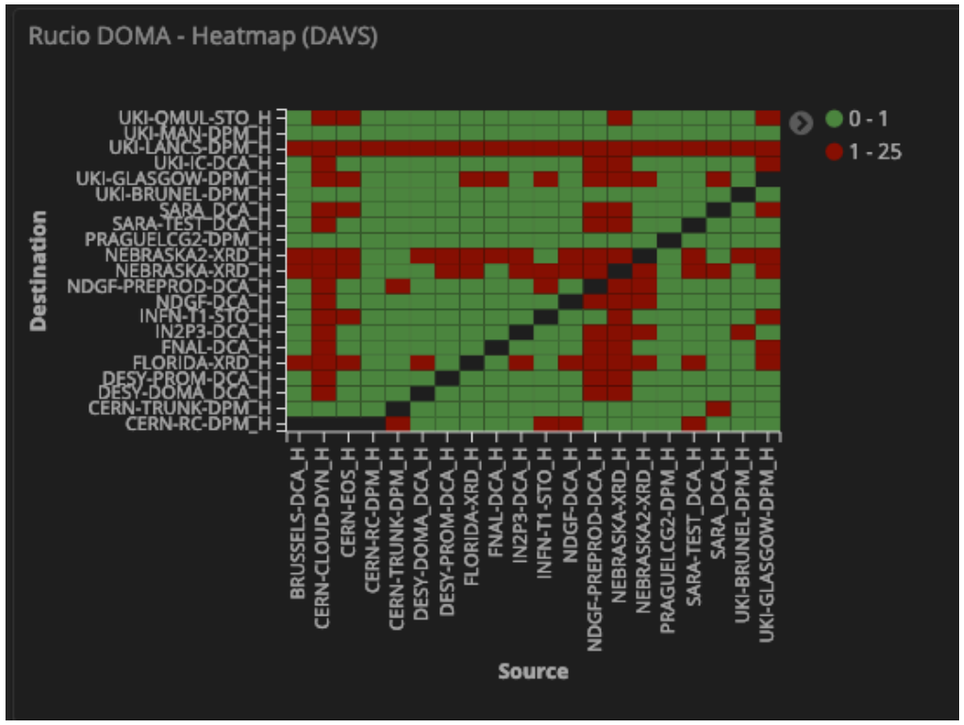
Token-based AuthN/Z in DOMA TPC WG

Bearer tokens used for authorization and delegation in HTTP third-party transfers across WLCG storage elements

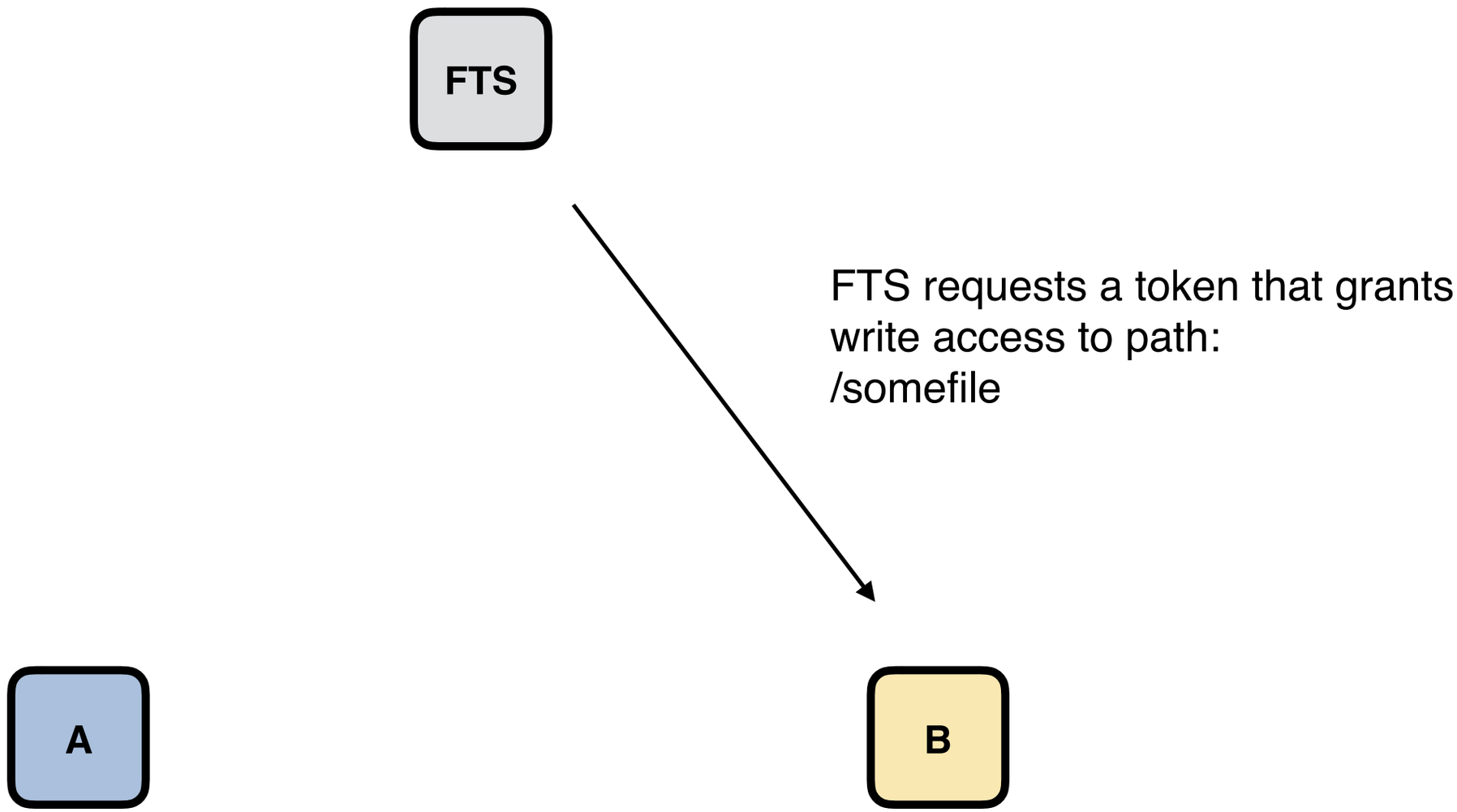
AuthZ already working fine for most SEs: dCache, DPM, StoRM, XRootD

FTS exchanges a VOMS proxy with an **SE-issued** authorization token, which grants (a subset of) the privileges granted by the VOMS proxy on such storage element

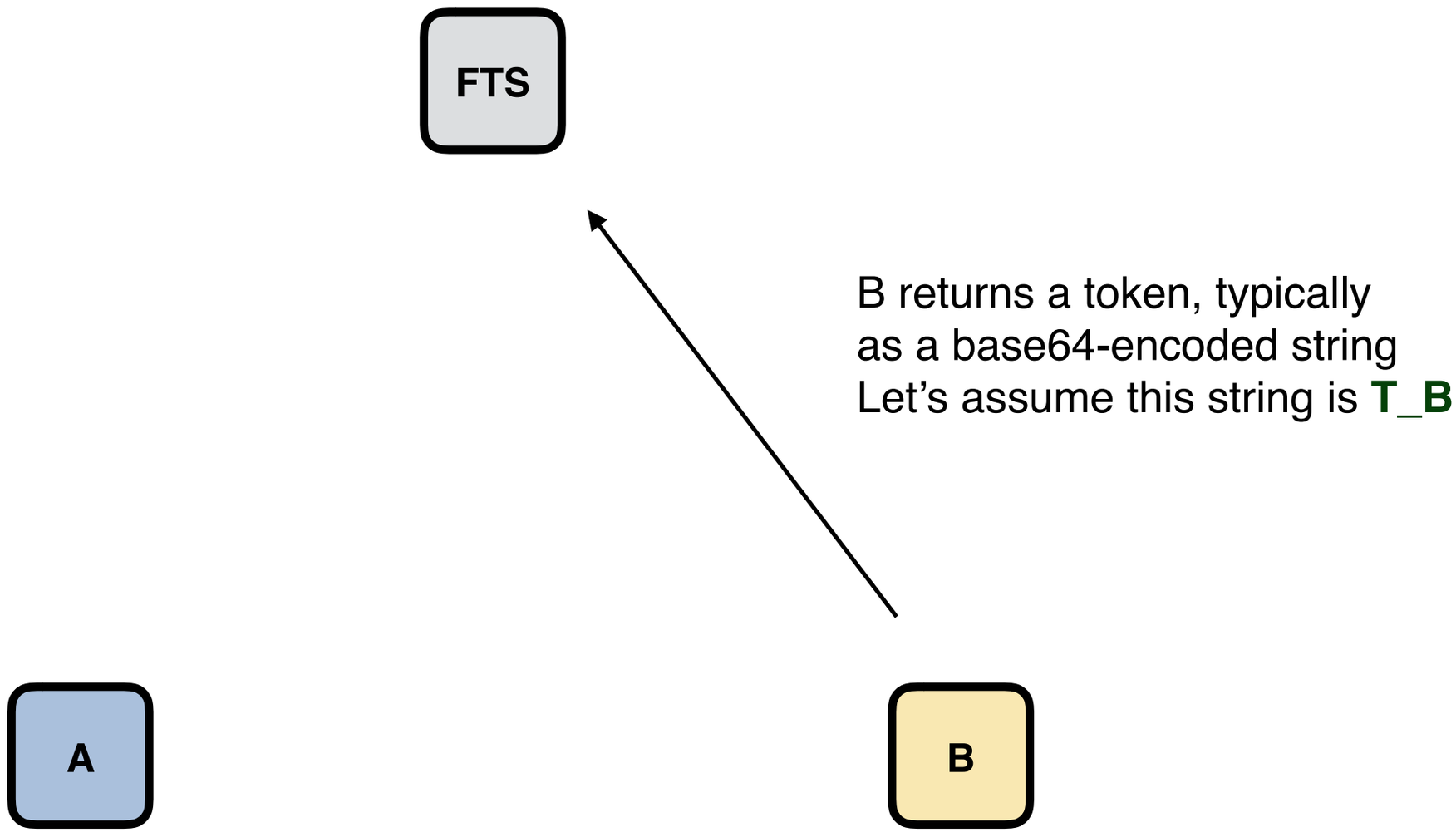
Agreement to converge on a common OAuth-based flow to request tokens



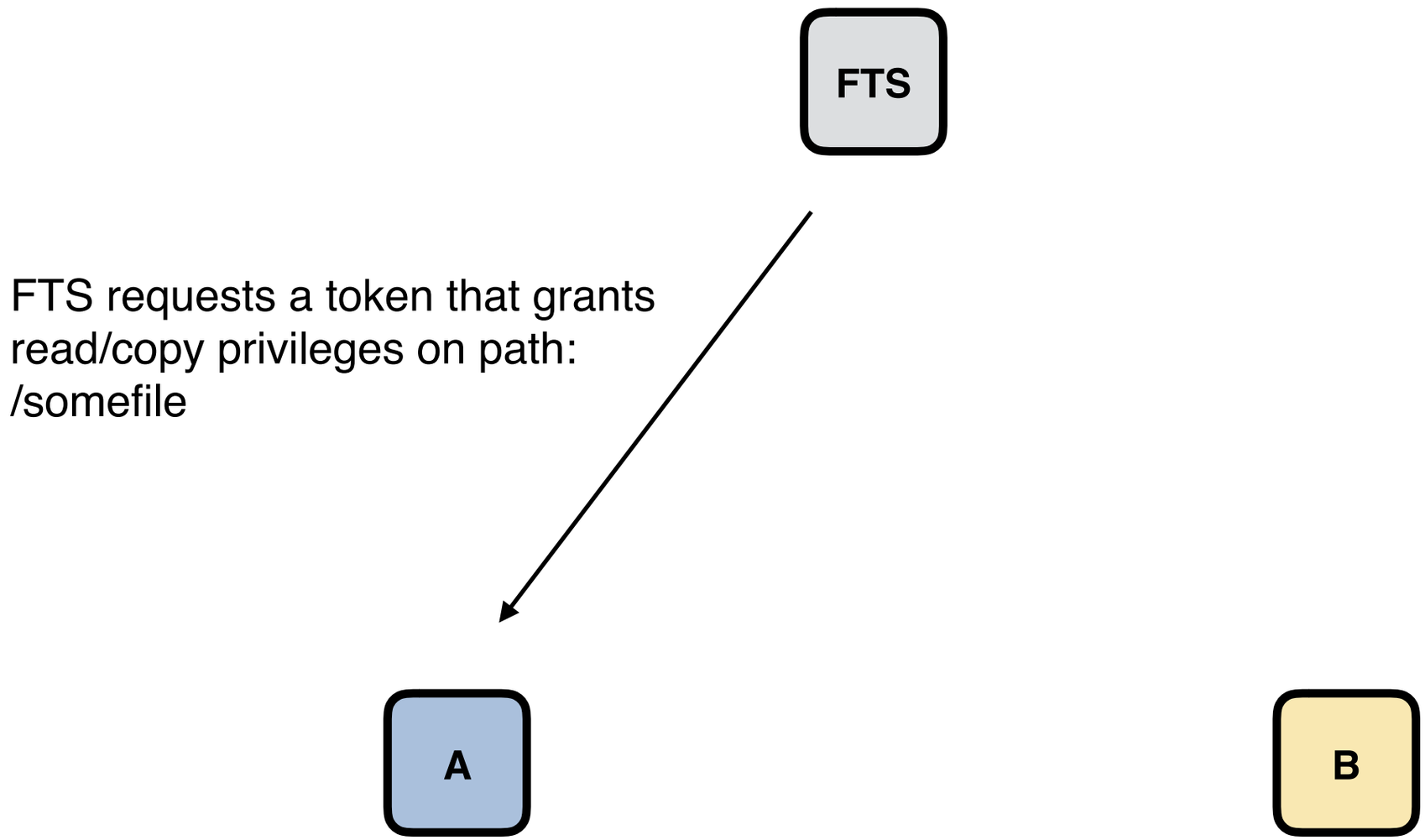
Token-based delegated AuthZ example



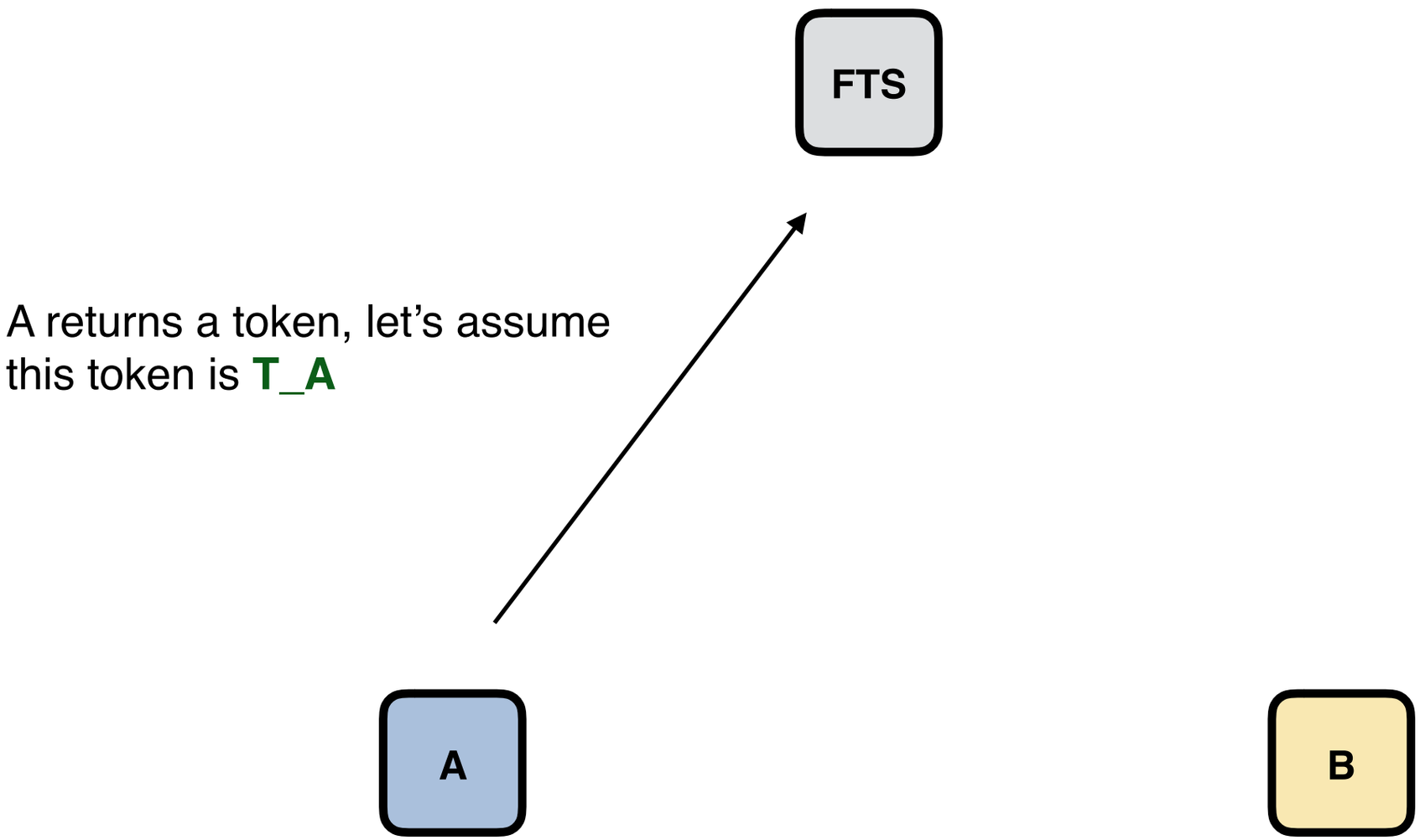
Token-based delegated AuthZ example



Token-based delegated AuthZ example

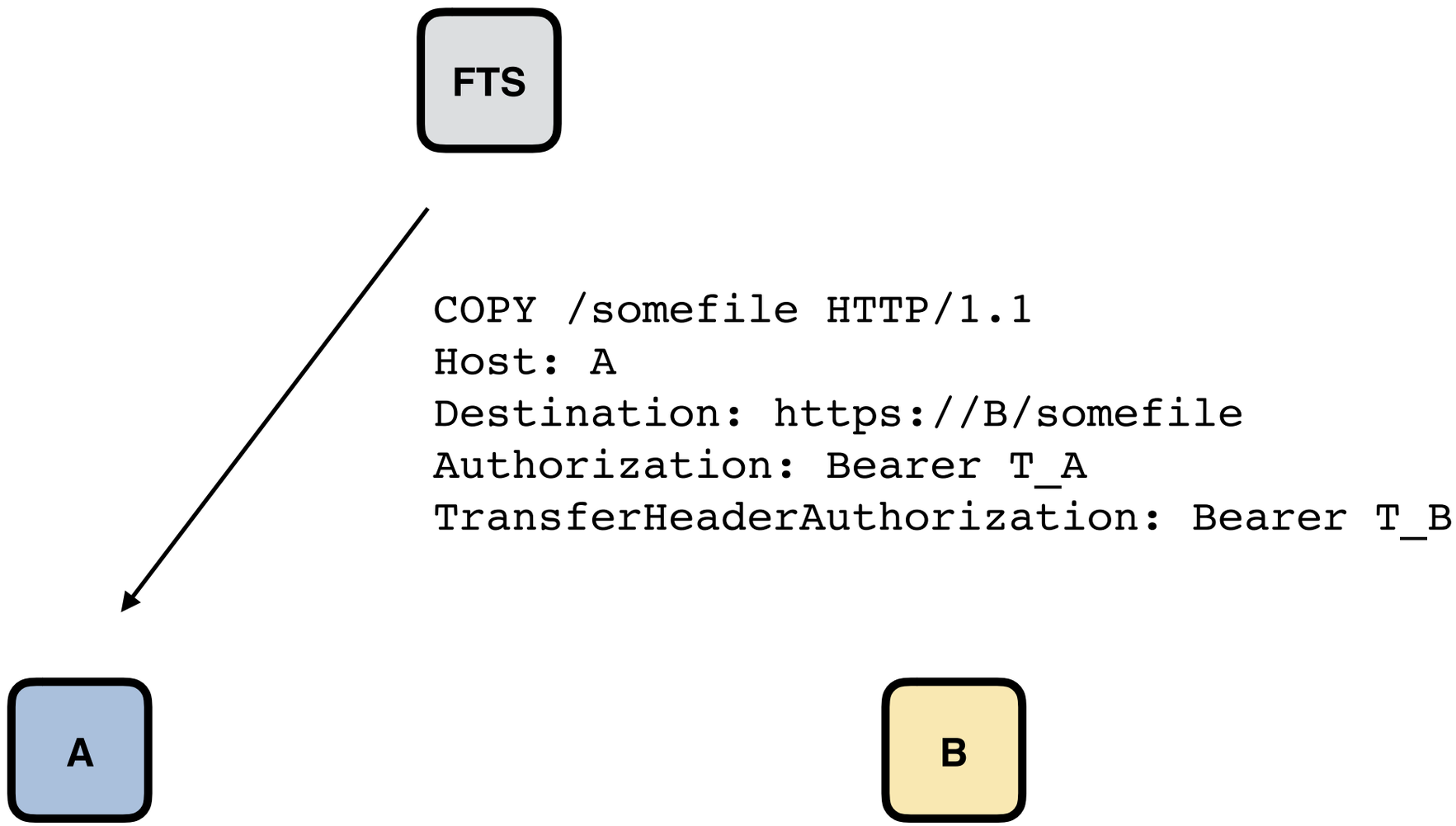


Token-based delegated AuthZ example



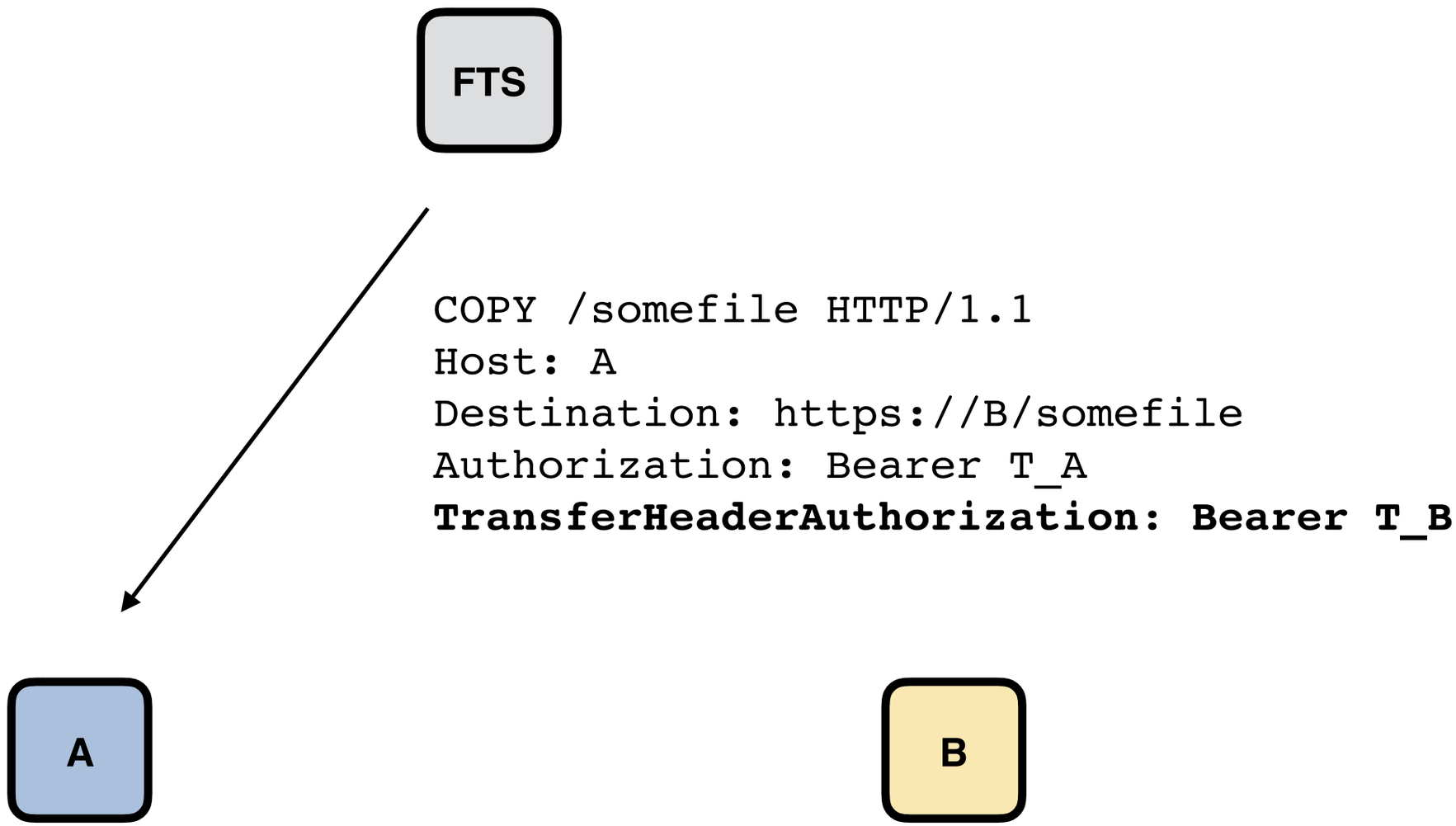
Token-based delegated AuthZ example

FTS can now request a TPC
from A/somefile to B/somefile

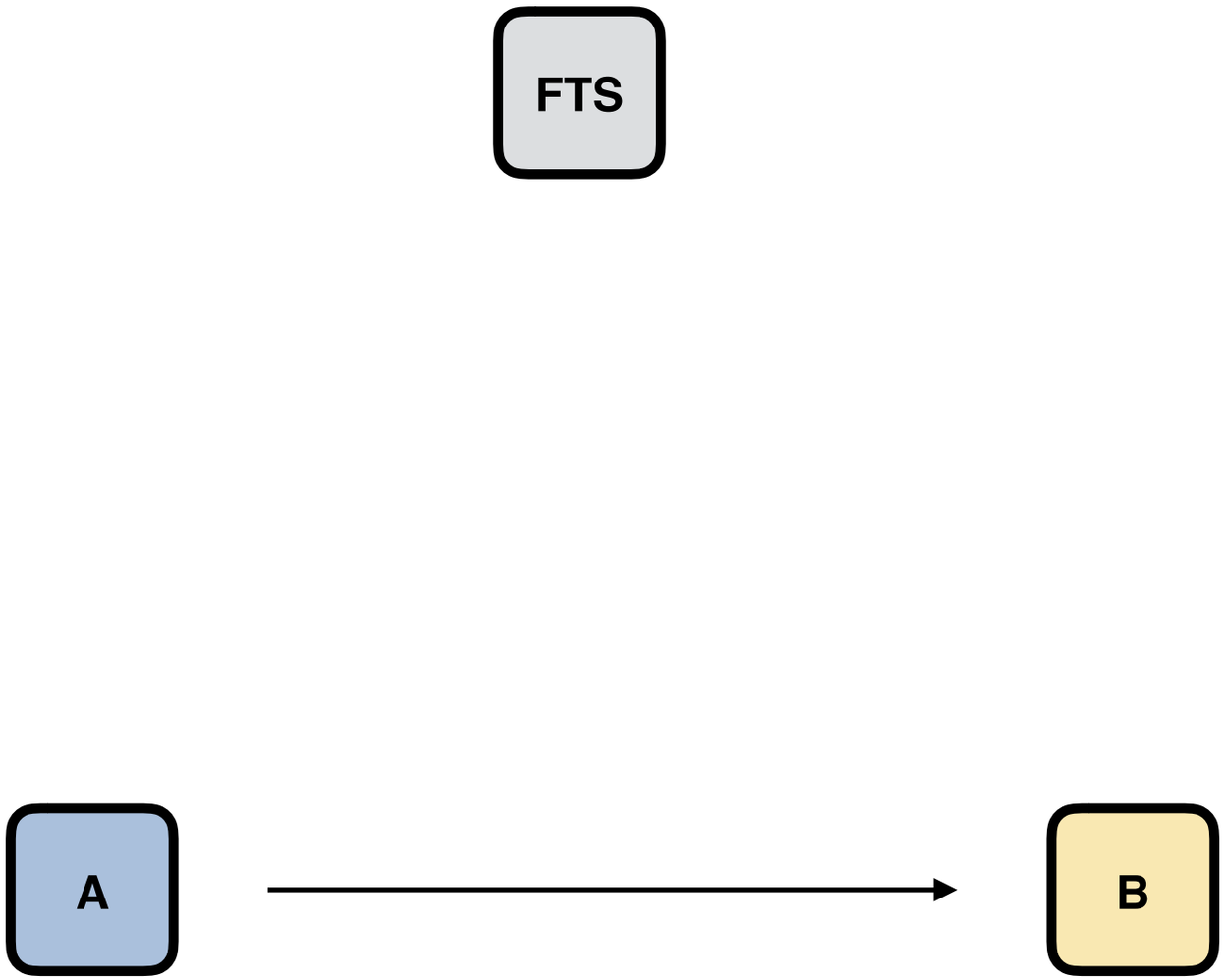


Token-based delegated AuthZ example

The protocol provides a way to request that certain headers in the COPY request are included in related transfer requests: all headers in the copy request starting with **TransferHeader** will be copied in the transfer request without such prefix.



Token-based delegated AuthZ example



Service-issued tokens vs VO-issued tokens

Service-issued tokens (like the ones used in DOMA TPC currently) only grant privileges on the service that issued the token

VO-issued tokens, OTOH, will grant access to all resources that trust the VO authorization server

- in a way similar to how VOMS attribute certificates grants access to all VO resources

In the future we will likely see both types of tokens used, and token translation services to translate between the two

- as it happens today at SEs to translate between VOMS ACs and an SE-issued tokens in DOMA TPC

INDIGO Identity and Access Management service

Flexible authentication support

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

Account linking

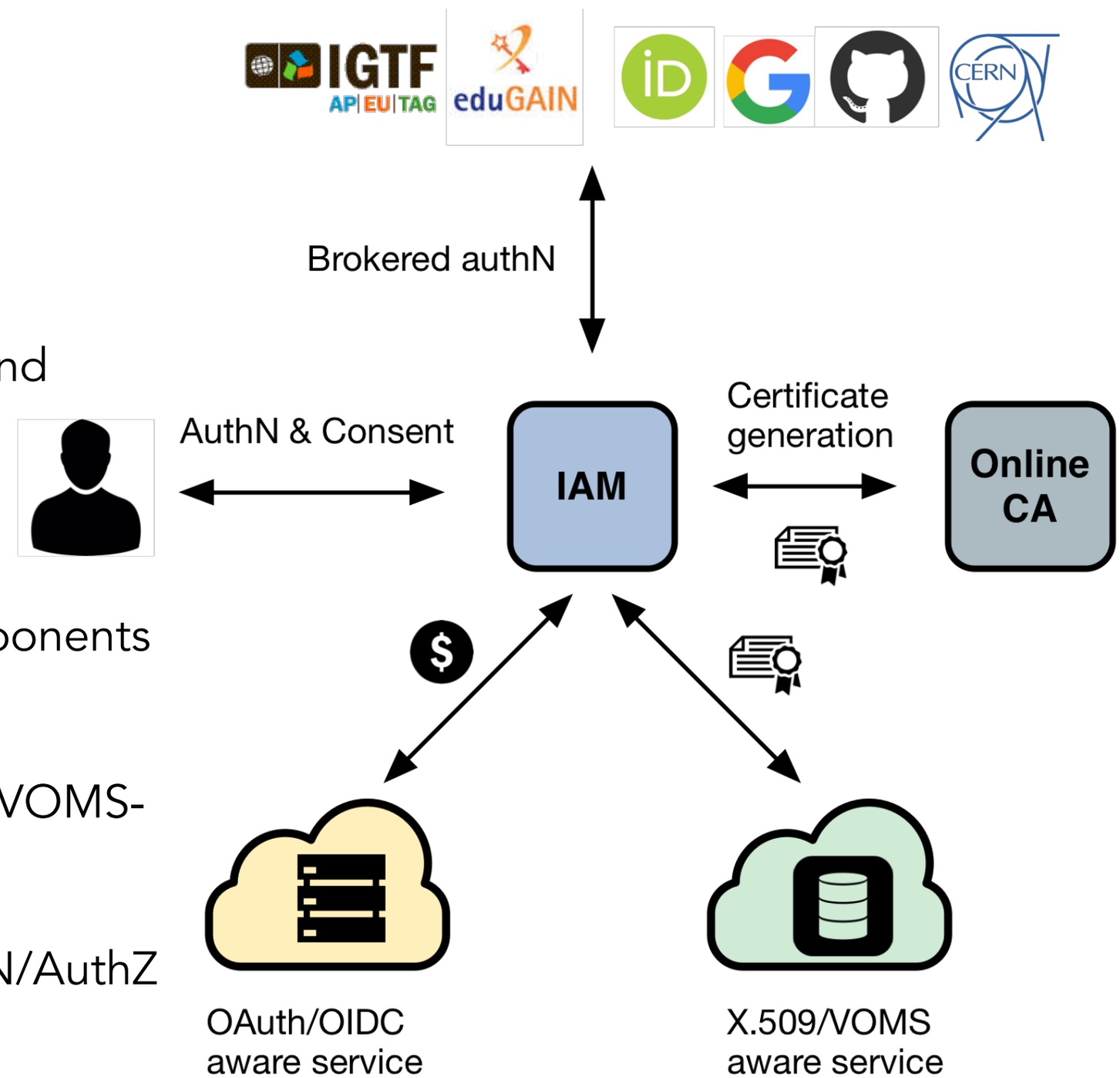
Registration service for moderated and automatic user enrollment

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

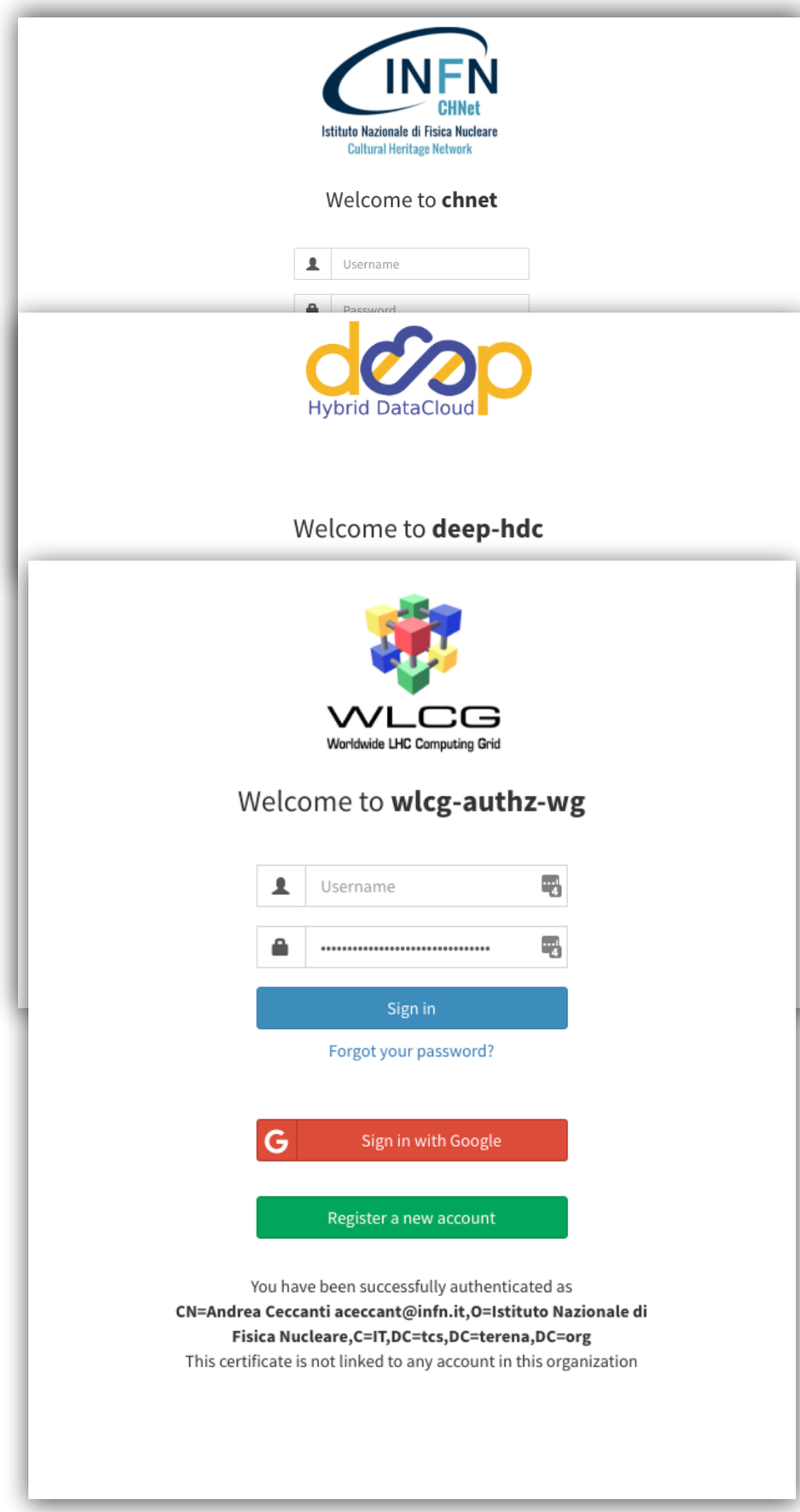


IAM deployment model

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

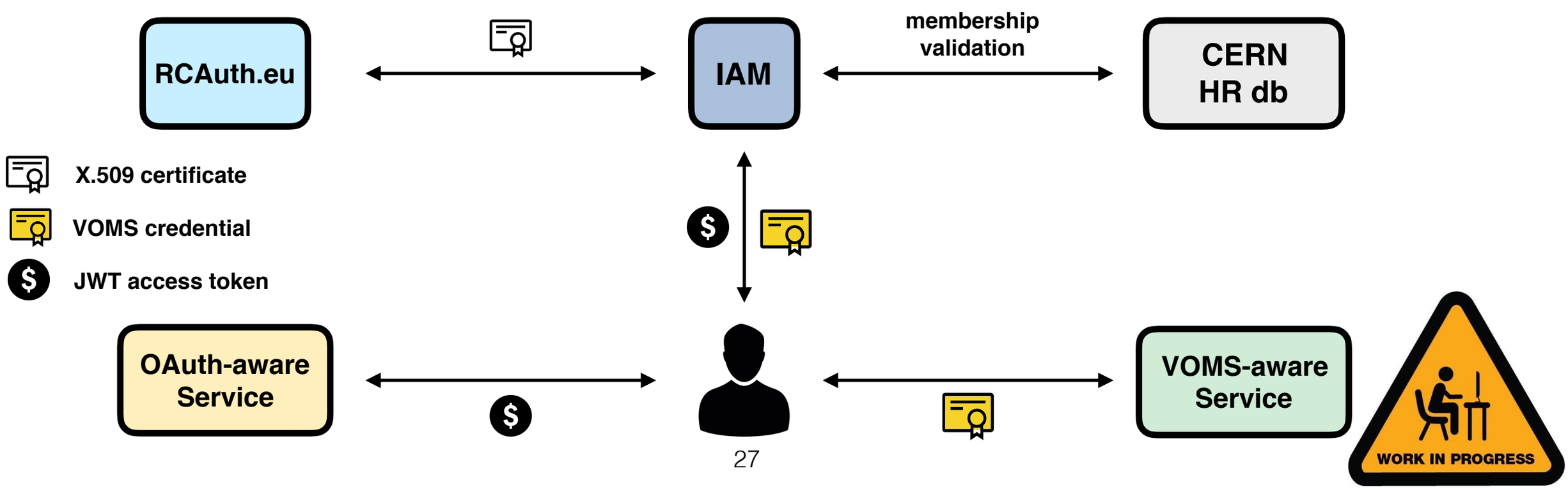


Seamless transition from an X.509-based AAI

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

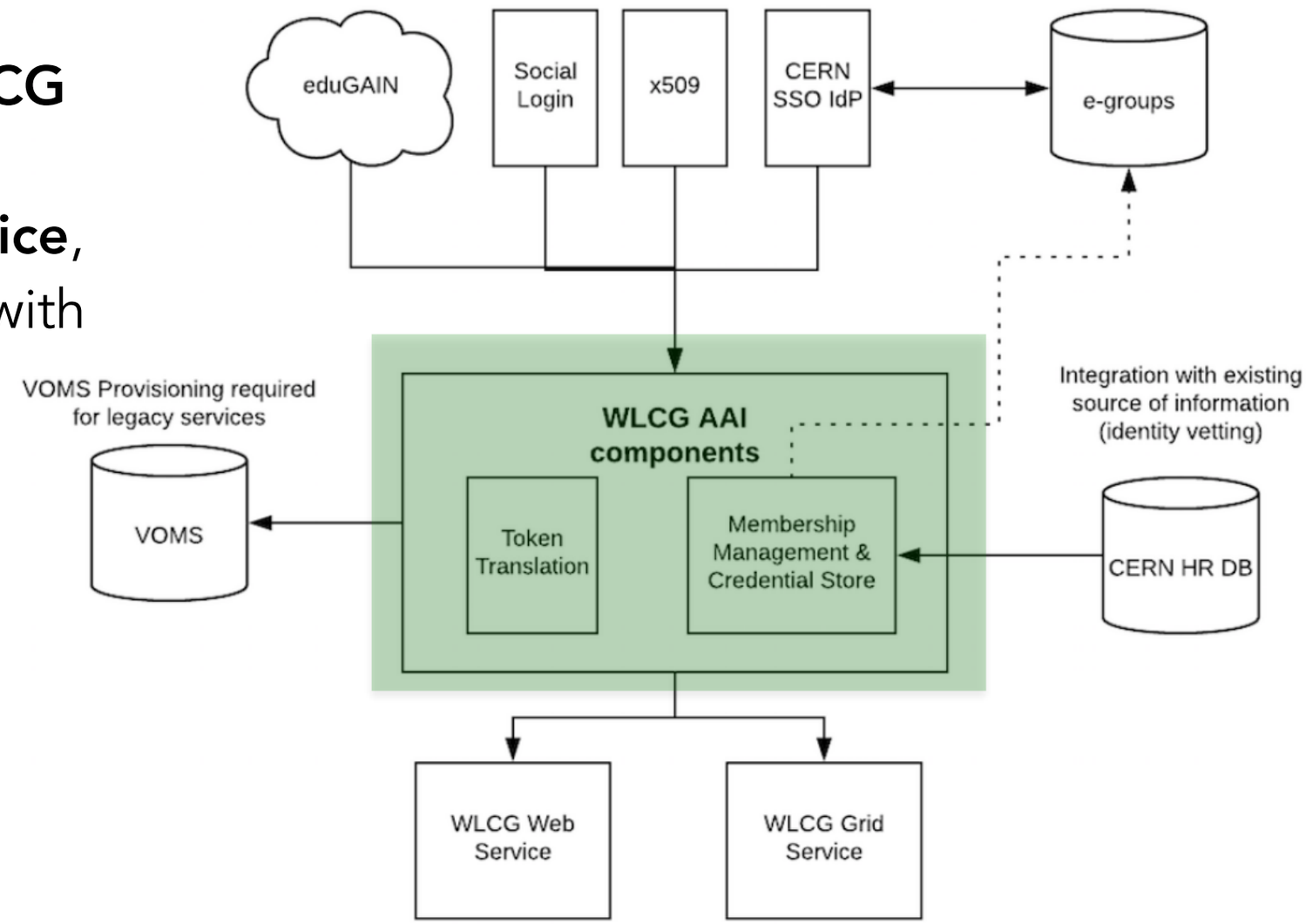


The WLCG Authorization WG

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**

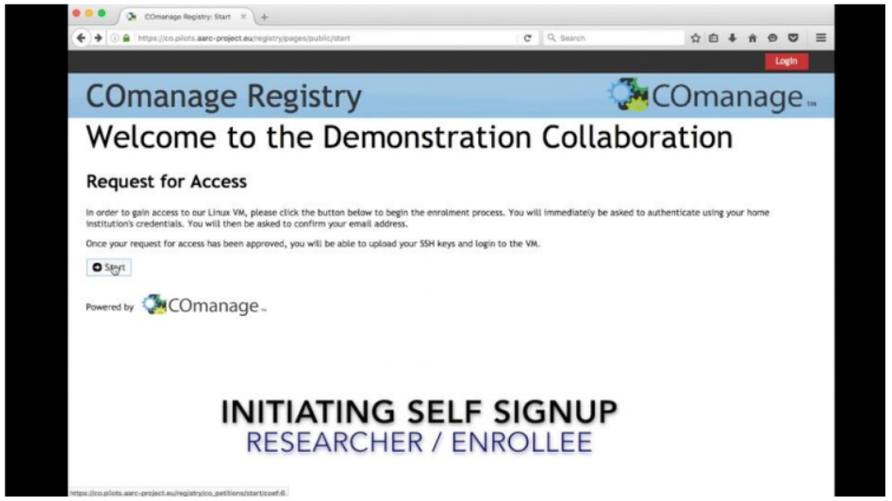


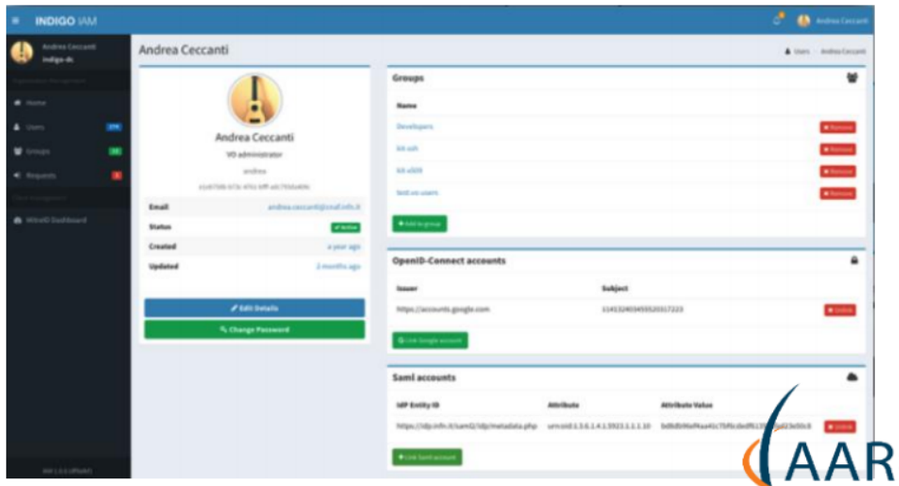
The WLCG Authorization WG

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

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





WLCG Worldwide LHC Computing Grid

WLCG AuthZ WG

AARC

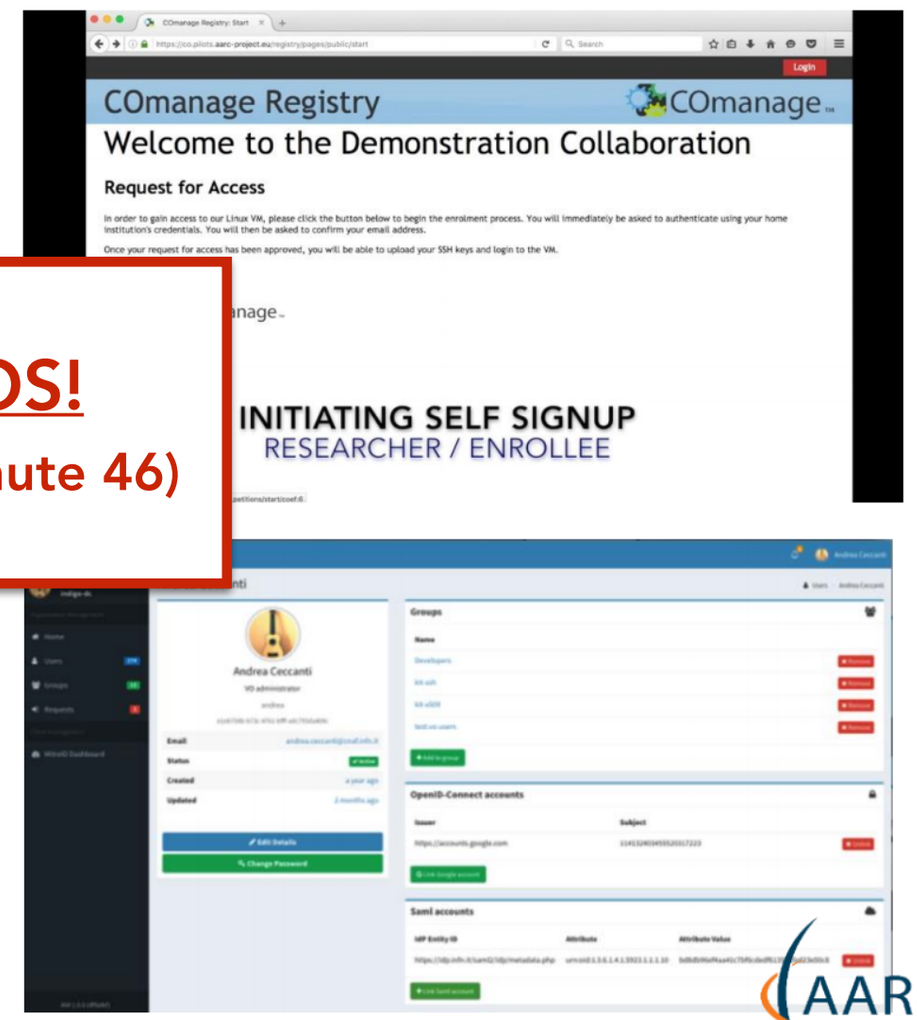
14

The WLCG Authorization WG

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

AAI Pilot Projects

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



See DEMOS!

(IAM starts at minute 46)

WLCG AuthZ WG

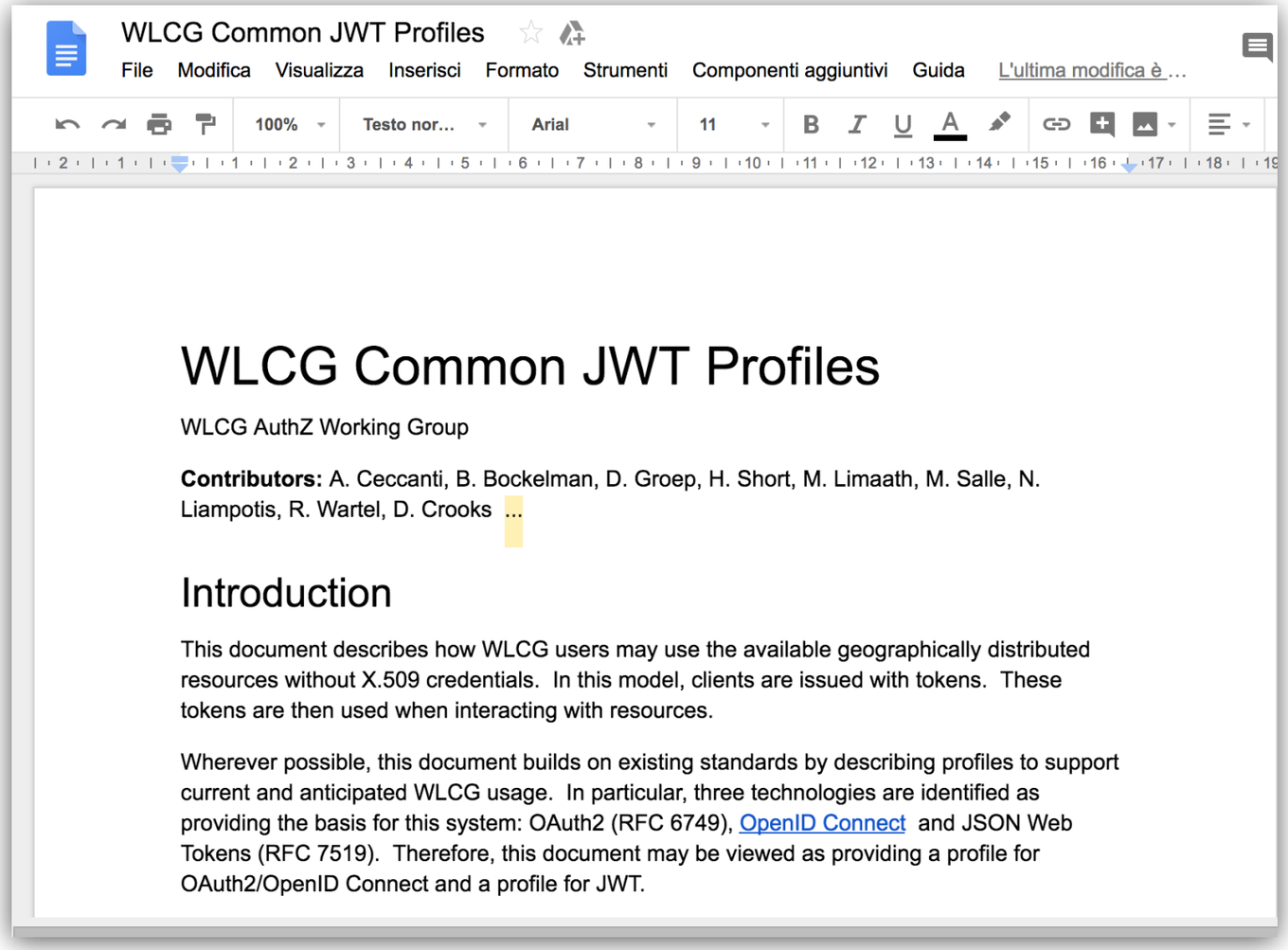
14

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 …

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?

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

WLCG AuthZ WG Demos: https://indico.cern.ch/event/791175/attachments/1806605/2948665/demos.mp4 (IAM starts at minute 46)

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

Contacts:

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

Backup slides

Integrating non-HTTP services

HTTP services are naturally supported by OAuth/Open ID Connect

What about non-HTTP services?

Use token translation:

- introduce an intermediate token-translation service that trusts the central VO authorization server and can translate VO-issued tokens to other types of credentials understood by the target service

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:

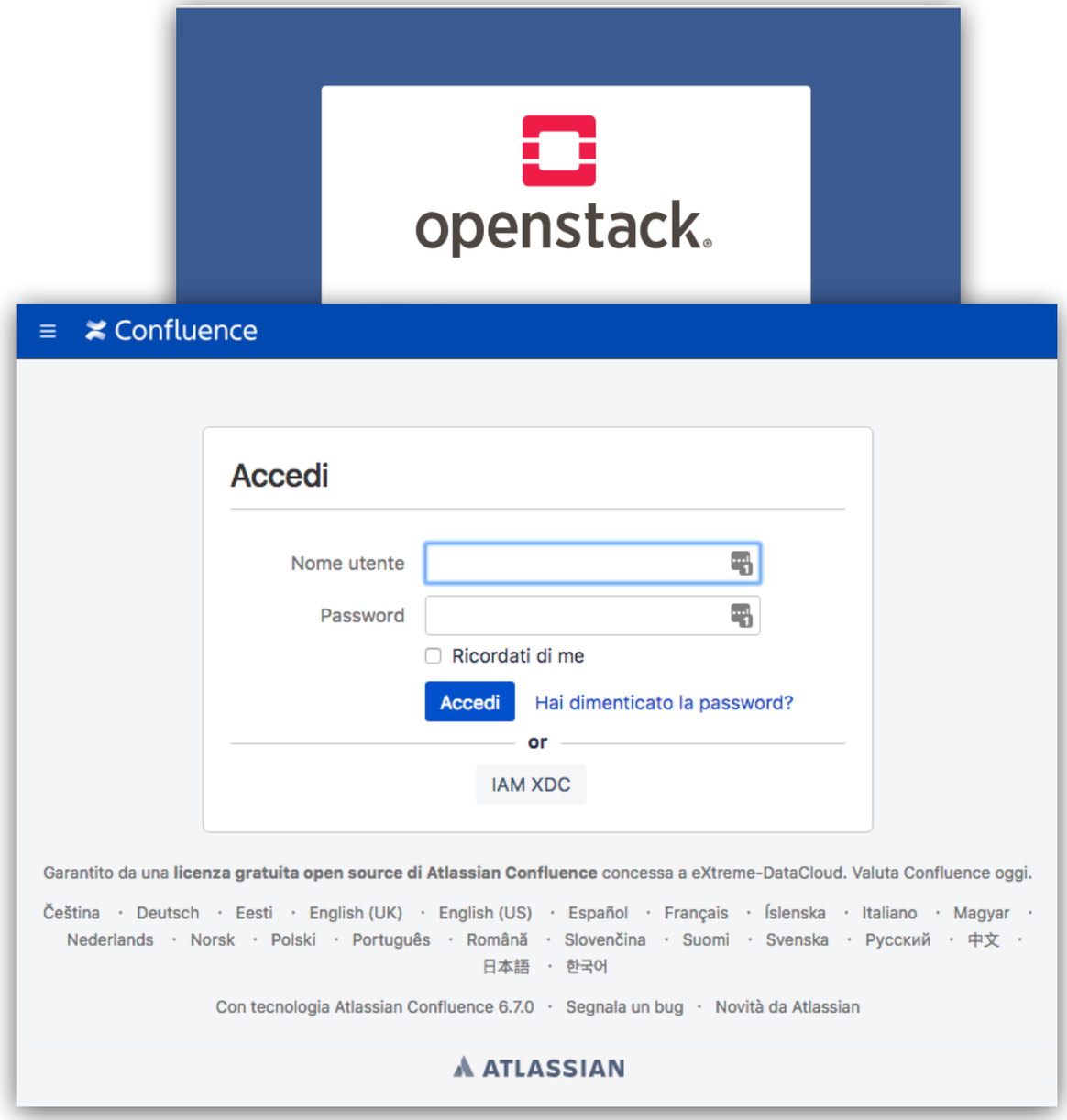
- Openstack
- Atlassian JIRA & Confluence
- Kubernetes
- Moodle
- Rocketchat
- Grafana
- JupyterHub

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

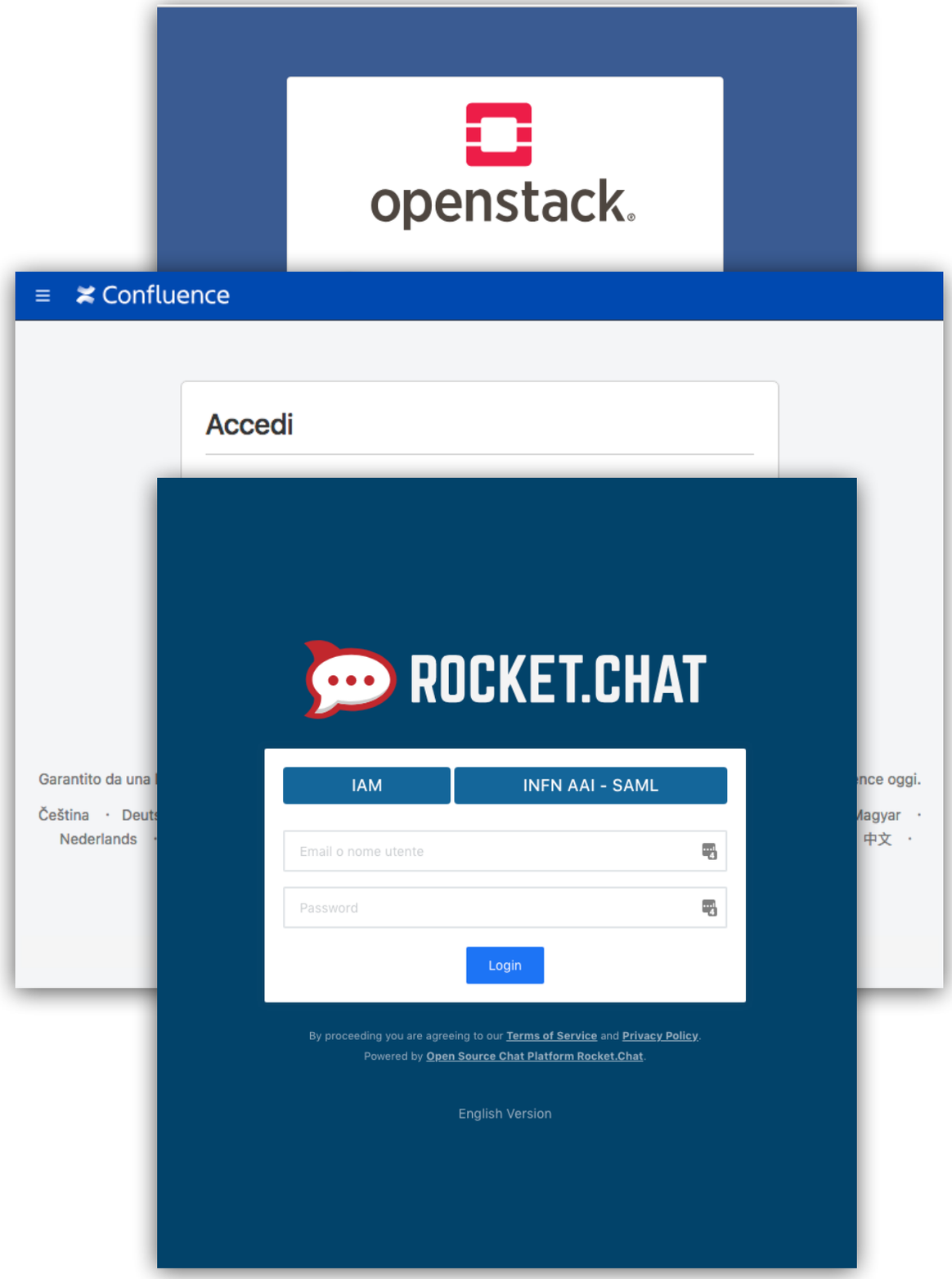


Easy integration with services

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



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

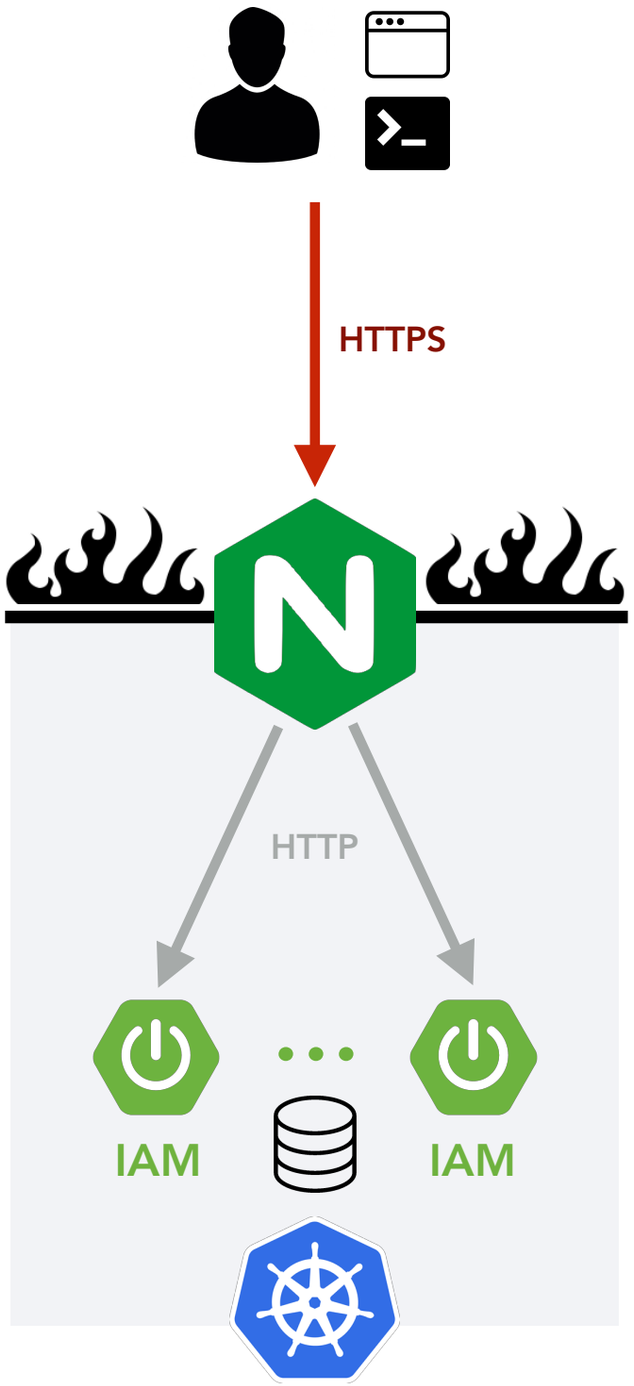
- all state persisted in the database

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

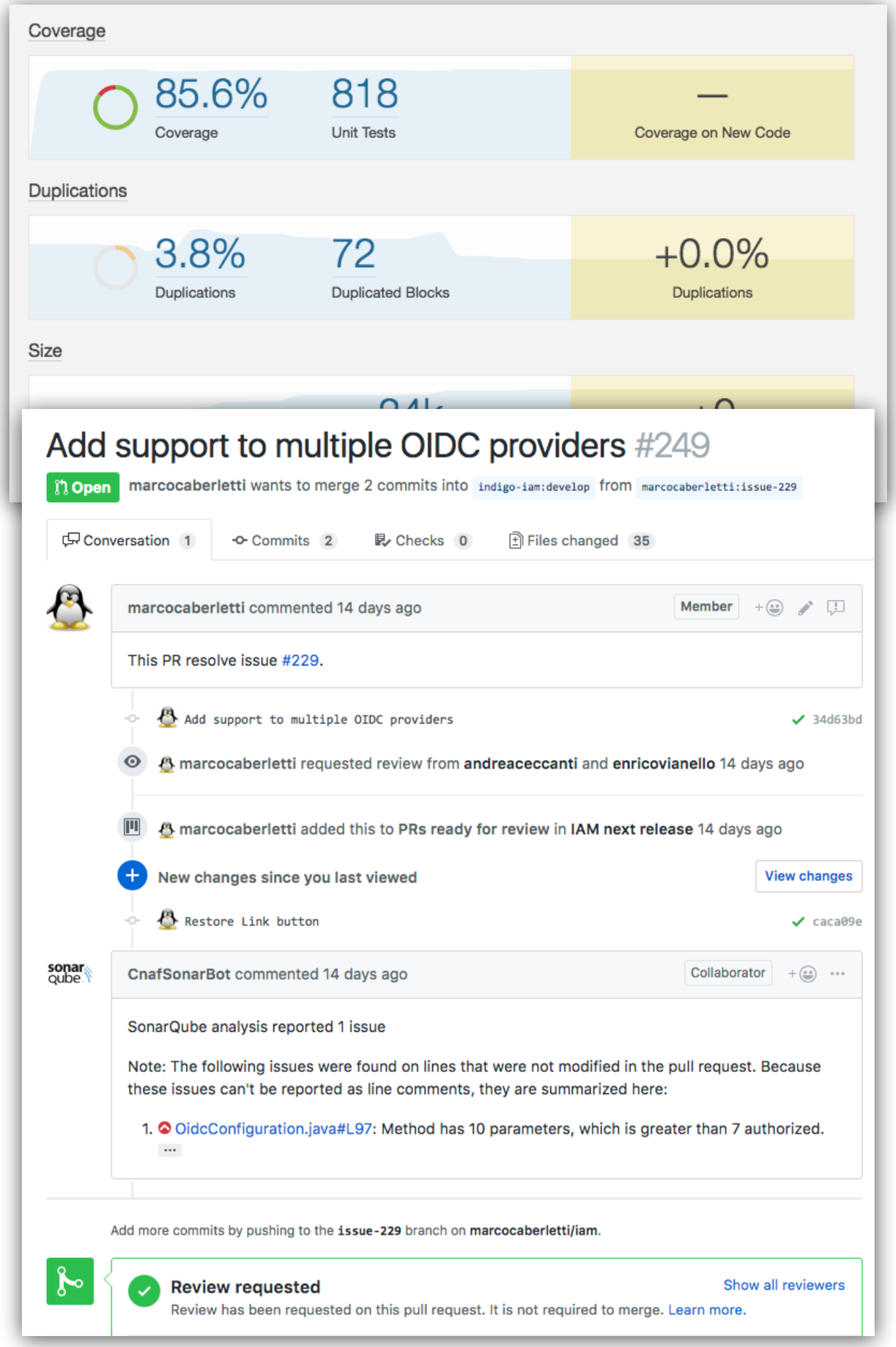
Open, **test-driven** development process

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