Evolving INDIGO IAM towards the next challenges

Enrico Vianello (INFN-CNAF)
enrico.vianello@cnaf.infn.it

CHEP 2024 - October 19-25th, 2024

INDIGO IAM

INFN CNAF
Istituto Nazionale di Fisica Nucleare

Topics

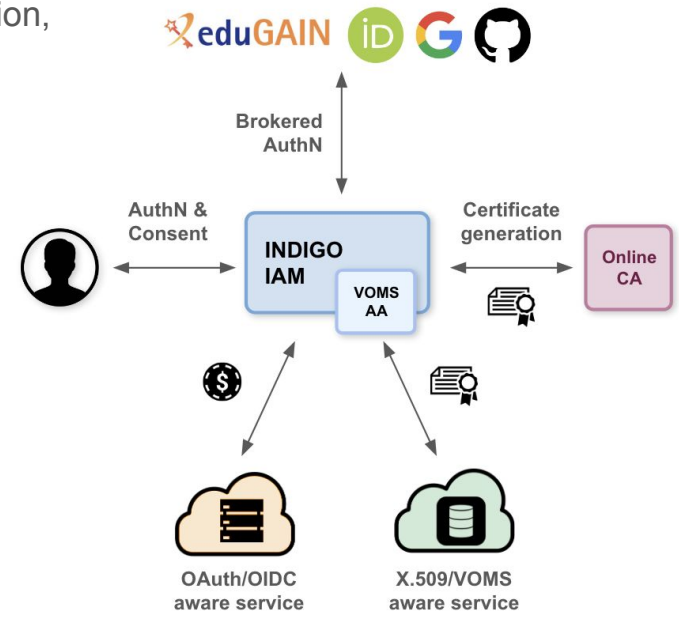
- INDIGO IAM
  - Brief service introduction, funding projects & its core technologies
- Where we are
  - Recent releases and relevant upgrades
  - Current development targets
- (In progress) IAM evolutions
  - Migration to Spring Authorization Server
  - New React-based Dashboard
  - Support for Multi-Factor Authentication
  - Open Policy Agent as engine for scope policies
  - Access tokens not stored on database

INDIGO IAM

INDIGO Identity and Access Management Service

"provides a layer where identities, enrollment, group membership and other attributes and authorization policies on distributed resources can be managed in an homogeneous way, supporting identity federations and other authentication mechanisms"

- Following **OpenID Connect** standard, exposes identity information, attributes and capabilities to services via **JWT tokens**
- Supports **multiple authentication mechanisms**
  - SAML, X.509, OpenID Connect (OIDC), local users
- Provides a **registration service** for moderated and automatic user enrollment (can be optional)
- Supports **account linking**
  - social and institutional SAML or OIDC accounts
  - x509 personal certificates and/or generated through an Online CA
  - SSH RSA keys
- Enforces **AUP acceptance** (can be optional)
- **Integrates easily** with ready-to-use components thanks to OpenID Connect/OAuth
- Can integrate existing **VOMS**-aware services



Funding projects & community

First developed in the context of the **INDIGO DataCloud** project, funded by the European Commission under the Horizon 2020 Programme → *first IAM v0.3.0 was officially released in 2016*

Selected by the **WLCG management board** to be the core of the future, token-based WLCG AAI

INFN-CNAF Software Development team currently works in collaboration with a community that includes also developers and IT people from **STFC** and **CERN**

➔ periodic Community Meetings and Hackathons (next on Nov. 27-28 in Orsay)

INFN commitment for the foreseeable future, with the current support of several Italian and European projects:

Core technologies

IAM is a **Spring Boot** application

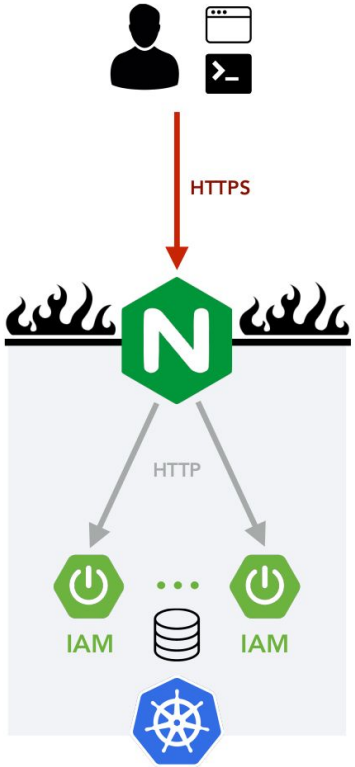
- OIDC/OAuth 2.0 implementation currently based on the MitreID Connect
- typically deployed behind an **NGINX**
- stores data in a **MariaDB/MySQL** database

Horizontally scalable

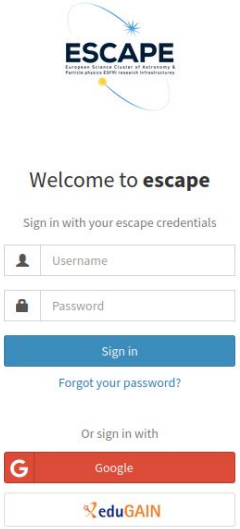
- sessions and external caching stored into **Redis** (or fully disabled)

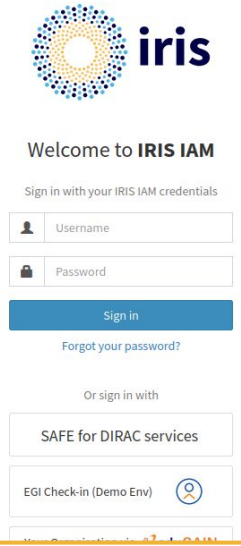
Typically deployed as a **containerized** service on top of **Kubernetes**

- autoscaling, zero downtime rolling updates



INDIGO IAM deployments - some examples





~ **20 instances inside CNAF** for internal purposes (INFN Cloud, CNAF Cloud, INFN T1 services, etc.) and scientific collaborations (ILDG, Belle-II, HERD, JUNO, etc.)

at least **4 instance at CERN** for LHCb, ATLAS, CMS and ALICE experiments and other instances for VO management (e.g. dteam)

1 instance at STFC for IRIS project

3 instances at IN2P3 for MesoNET, EURO-LABS, GRANDMA projects

Where we are

Latest releases (since CHEP 2023)

Less monolithic and more frequent releases

May **2023**: **v1.8.2** → introduced admin scopes, moved to Spring-Boot v2.6.14

December **2023**: **v1.8.3** → hashed AT values on db and more db enhancements

March **2024**: **v1.8.4** → ui customizations, other db enhancements

June **2024**: **v1.9.0** → disabled clients, AUP re-sign and sign-on-behalf, AUDIT improvements, more info from SCIM endpoints (authorities, attributes and managed groups), clients last token issued info added

August **2024**: **v1.10.0** → AUP management and notifications enhancements, automatic groups enrollment on registration, statistical endpoint

September **2024**: **v1.10.1** → Mainly a bug fixing release, AngularJS latest version

October **2024**: **1.10.2** → CERN lifecycle logic fixes

Current development targets

- **Improve auditing**
- **Superseded obsolete dependencies**
  - MitreID → Spring Authorization Server
  - AngularJS → React JS
- **Improve usability** for users & admins
- **Scalability and performances improvements**
  - Access tokens not stored on database
  - Dedicated garbage collector service
  - Scope Policies evaluated with Open Policy Agent (OPA)
- **Interoperability focus**
  - Support OIDC Federations
  - Improve conformance with AARC BluePrint Architecture and its guidelines
- **Security**
  - Add Multi-Factor Authentication (MFA)

Next IAM evolutions

Migrate to Spring Authorization Server

Spring Authorization Server is a framework, built on top of **Spring Security,** that provides a secure, lightweight and customizable foundation for building an **OAuth 2.1** and **OpenID Connect 1.0** Authorization Server implementation.

Why?

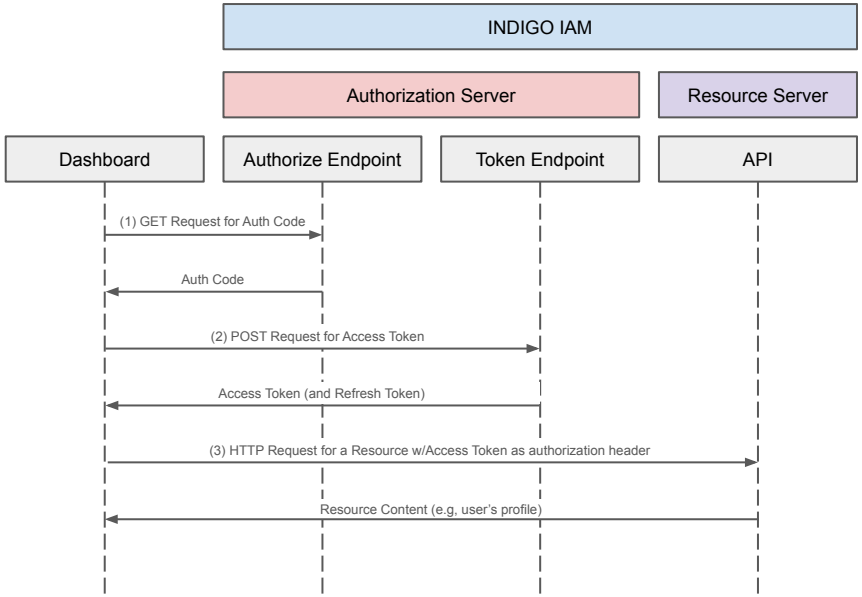
- We still rely on a forked and self-maintained version of MitreID Connect library which has no substantial support/evolution since few years
- It's a natural evolution of the current architecture Java/Spring based
- Long-term support and easier maintainability
- Better OIDC/OAuth standards compliance
    - Compliance with OAuth 2.1 standard

New IAM dashboard: motivations

- Remove AngularJS (EOL) and JavaServer Pages
- Use a modern and lightweight rendering framework (**React**)
- Benefit from using a modern **HTML5** / **TypeScript** / **CSS** development stack
- **Decouple** the frontend code from the OIDC/OAuth2 implementation
- Handle AuthN/AuthZ via OpenID Connect and OAuth2 frameworks
- Enhance **customizability** for different organizations

New IAM dashboard implementation

- AuthN/AuthZ responsibilities managed by the web application
  - OAuth2 **Authorization Code flow** (RFC6749) w/ **PKCE**[1] extension (RFC7636)
- Requests to the INDIGO IAM endpoints authenticated via the obtained **JWT access token**
  - INDIGO IAM plays both the roles of **Authorization Server** and **Resource Server**



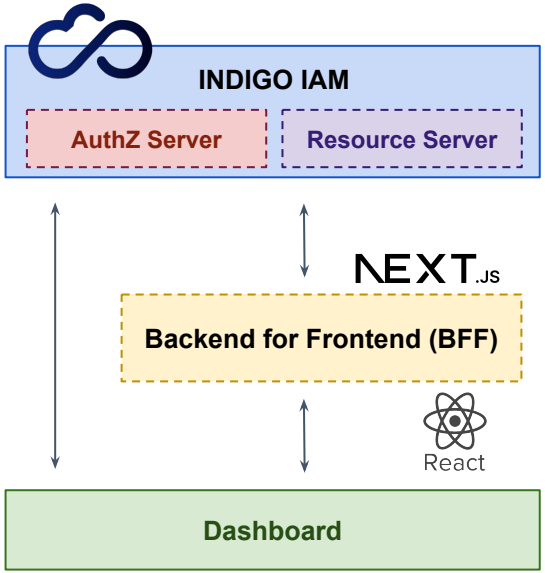
OAuth2 Authorization Code flow (PKCE is not shown in figure)

[1] **P**roof for **K**ey **C**ode **E**xchange

New IAM dashboard implementation

We are following the **Backend For Frontend (BFF) pattern** for security reasons

- The BFF interacts with the authorization server as **a confidential OAuth client**
- The BFF **handles all OAuth2 responsibilities** and forwards requests to the Resource Server after adding the appropriate access token
  - **no OAuth tokens exposed** to the browser
  - a **cookie-based session** keeps OAuth tokens secure from the JavaScript application
- Rendering and computations completely run on the backend server exposing only the final HTML content
- We are using **Next.js**, an advanced web development framework, for server-side rendering

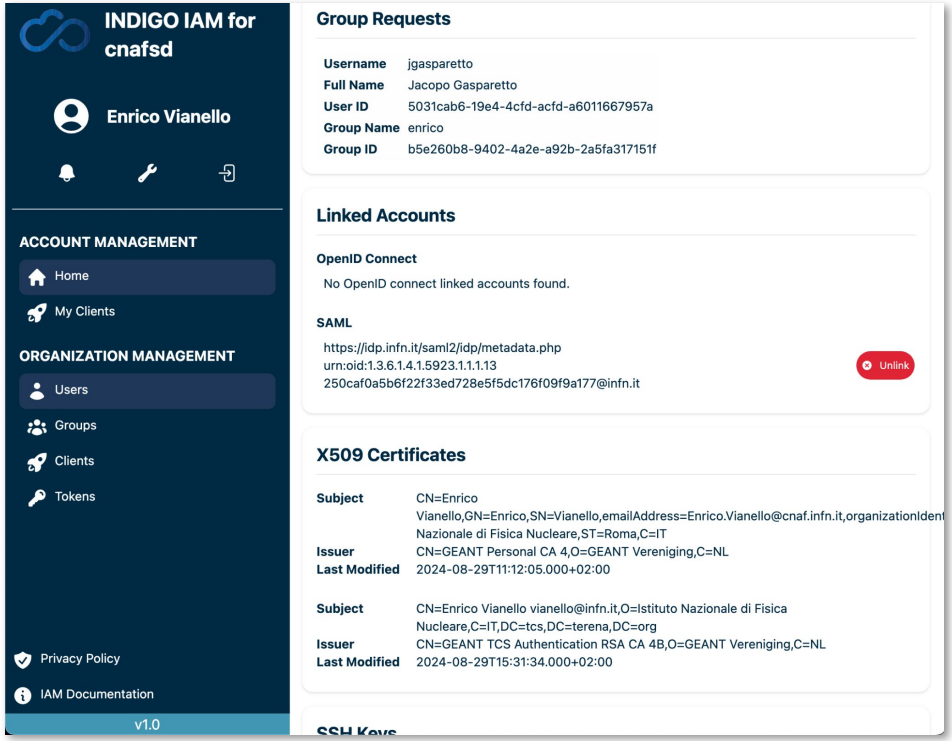


New IAM dashboard

- Simple and lightweight
- Deployed as a **Docker image**
- Highly **scalable**

Currently a demo version is deployed on our development Kubernetes cluster using ArgoCD

GitHub Source



Homepage example

Multi-Factor Authentication (MFA)

What's done

- Authenticator app working for authentication with local credentials
- Multi-factor settings menu on dashboard
- 2FA enabled by configuration

In progress

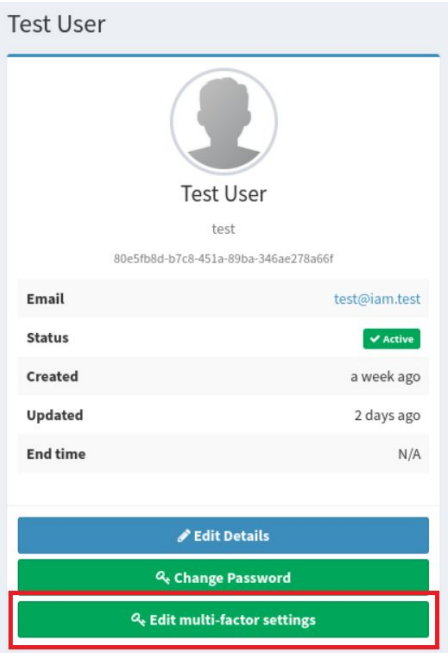
- Encryption and decryption of MFA secrets

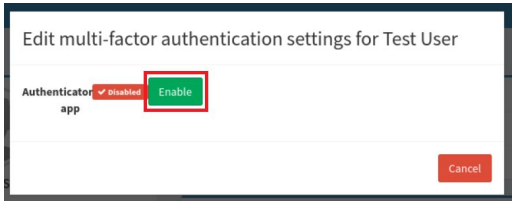
To Do

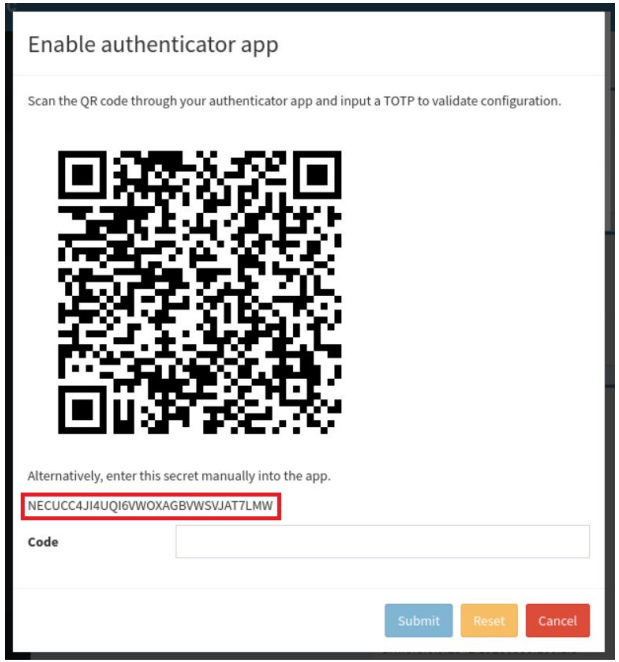
- Integrate 2FA when login with external identity providers
  - allow skipping IAM 2FA in case the user has already done it through the external IdP
- What's necessary to satisfy all the established best practices
  - e.g. allow IAM administrators to disable 2FA per user

Enabling 2FA for local credentials

UNRELEASED

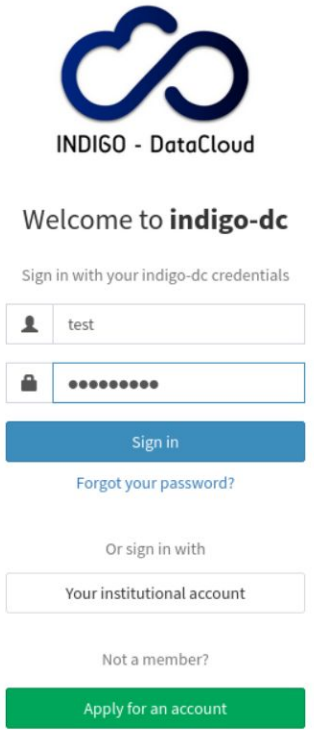


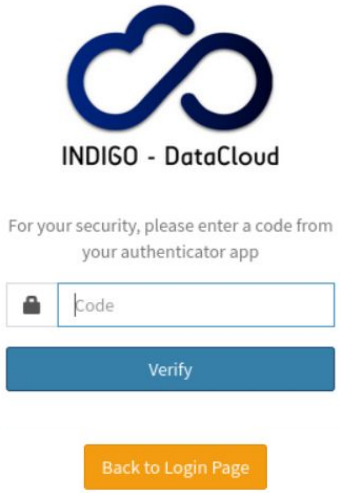




Login with 2FA enabled for local credentials

UNRELEASED

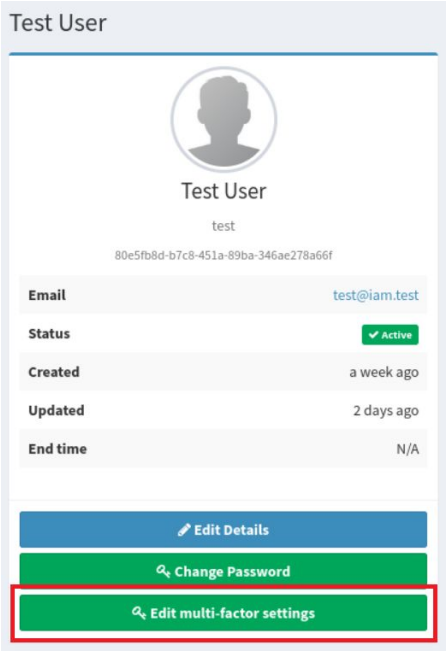


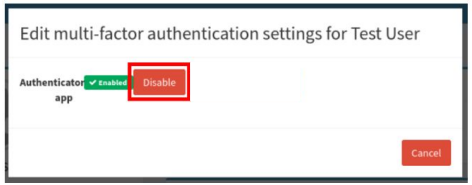


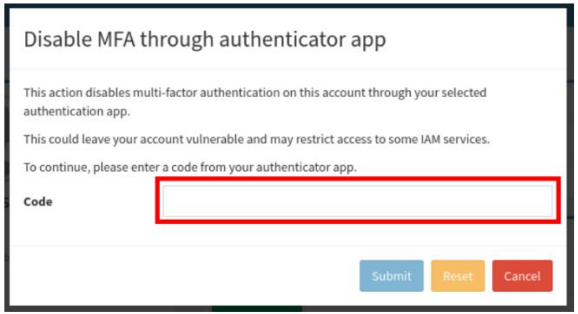
Authenticated!

Disabling 2FA

UNRELEASED







Integration with Open Policy Agent

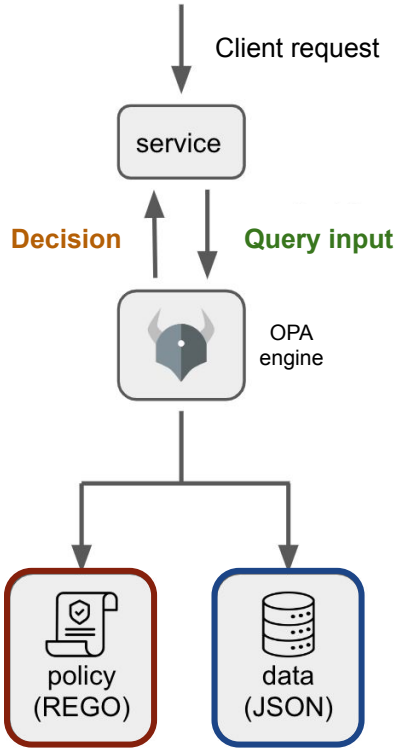
Open Policy Agent (OPA) is an open-source **authorization engine**

OPA is based on an high-level declarative language (**Rego**) that allows the definition of **policies as code**

Rego ensures low latency policy decisions, even with a large number of rules

A service which needs to take a policy decision can query OPA passing arbitrary structured data (JSON or YAML) as input. The OPA engine evaluates the query input against REGO policies and optional data.

An OPA decision is not limited to a simple allow/deny answer, but can include **arbitrary structured data as output**



Evaluation of scope policies with OPA

IAM Scope Policies provide a mechanism to **control access to token scopes**

- the list of requested scopes is filtered through policies applied to **users** or **groups**

Through proper rules defined through the REGO language, we implemented and evolved the current IAM policy-decision-point logic:

- e.g. policies are applied also to **clients** to support the OAuth client credentials flow (not bound to a user)

The definition of policies (in a data file) is backward compatible with IAM

The **`opa eval --profile`** command has been used to profile the scope policies:

- applying 10k scope policies through OPA took ~130ms while IAM reached the OIDC-Agent client timeout!

METRIC	VALUE
timer_rego_module_compile_ns	52170843
timer_rego_module_parse_ns	12619578
timer_rego_query_compile_ns	716752
timer_rego_query_eval_ns	129958182
timer_rego_query_parse_ns	750061

Access tokens not stored on database

Why?

- during latest data challenge we reached millions of access tokens stored on database
  - the most of them expired and slowly deleted (with performance loss)
  - not really a need if IAM is not acting as a Resource Server (API calls)

How to:

- validate the Access Tokens used to contact IAM API endpoints
  - we can cache the result with a proper eviction time (same as expiration datetime)
- persist revoked access tokens
  - it’s not a common use case in the end - OAuth2 protocol by design is built to “accept” the loss of an access token

Other coming things …

- Support **OpenID Connect Federations**
  - EOSC Beyond T8.1 "*Extending Capabilities of EOSC Core Services and Supporting EOSC Nodes*" is focused on OpenID Federations and AuthN methods beyond passwords
    - Initial release of the next generation of EOSC Core Services → **Aug 29, 2025**
- Conformance with **AARC BluePrint Architecture** and (more) guidelines
  - The AARC Blueprint Architecture (BPA) is a set of software building blocks that can be used to implement federated access management solutions for international research collaborations
  - AARC has guidelines and best practice recommendations to support the implementation of the Blueprint Architecture.
  - Following/implementing AARC guidelines will improve the interoperability between infrastructures

Conclusions

INDIGO IAM is a critical service widely adopted by many scientific communities. Our evolution roadmap includes:

- Migration to Spring Authorization Server
  - Go beyond the unsupported MitreID Connect library
  - Better compliance with OIDC / OAuth 2.1 standards
- Development of a new dashboard
  - Go beyond old AngularJS-based web user interface
  - Decouple frontend codebase from INDIGO IAM
- Support for Multi Factor Authentication
  - Partially implemented, to be released by the end of 2024
- Integration with Open Policy Agent as scope policies engine
  - Performance improvements and a good testbed for more applications
- No more Access Tokens stored in database
  - Performance improvements, release candidate expected by the end of 2024
- Support for OIDC Federations
- Conformance with AARC BluePrint Architecture and its guidelines

Thanks! Questions?

Contacts and references

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

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

For general information:

- OAuth 2.0: https://oauth.net/2/ and OAuth 2.1: https://oauth.net/2.1/
- OpenID Connect: https://openid.net/connect/
- OpenID Connect Federation: https://openid.net/specs/openid-connect-federation-1_0.html

Contacts:

- iam-support@lists.infn.it