# PUNCH AAI Updates

*Oliver Freyermuth*, Michael Hübner, Kilian Schwarz, TA6, Christoph Wissing

28th March, 2025


UNIVERSITÄT BONN

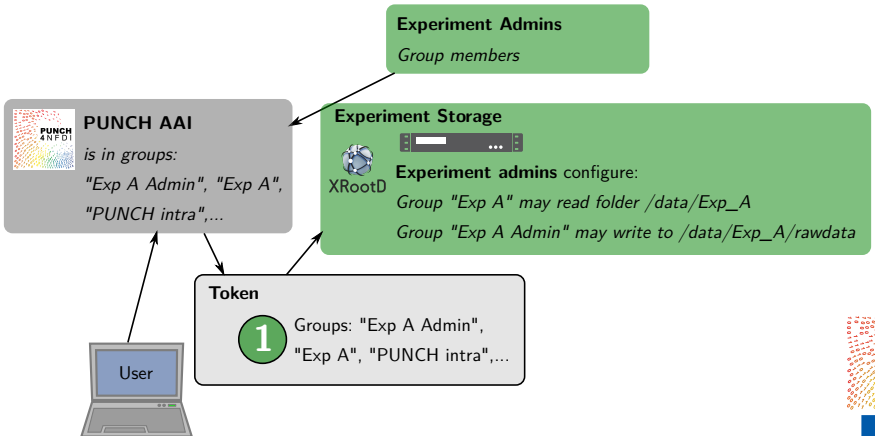
# AAIs in the PUNCH Communities

1. ‘PUNCH AAI’ (Helmholtz AAI) based on Unity IAM
   - Instance available since the start of the PUNCH4NFDI consortium
   - **Community AAI in IAM4NFDI** thanks to collaboration via FZJ & KIT
     $\Rightarrow$ required for Base4NFDI services (MultiCloud etc.) and offering NFDI-wide services
   - **Yet lacking some functionalities for token authorizations**
   - Lacks a policy engine (granular permissions)
   - Group management possible (can be delegated)
   - Successful development: Permissions in Indico can be controlled based on AAI groups
   - **Established development workflow exists** (direct developer contact by now)
2. ‘Indigo IAM’ (used by WLCG, Belle II, ILDG, SKA)
   - **Not one of the IAM4NFDI services**
   - Very compatible with the advanced WLCG-tools re-used within PUNCH4NFDI
   - Has all required functionalities for token authorizations and a policy engine
     Note: More elaborate policies may be required, see e. g. ILDG
   - **Unclear procedure if additional features need to be developed**

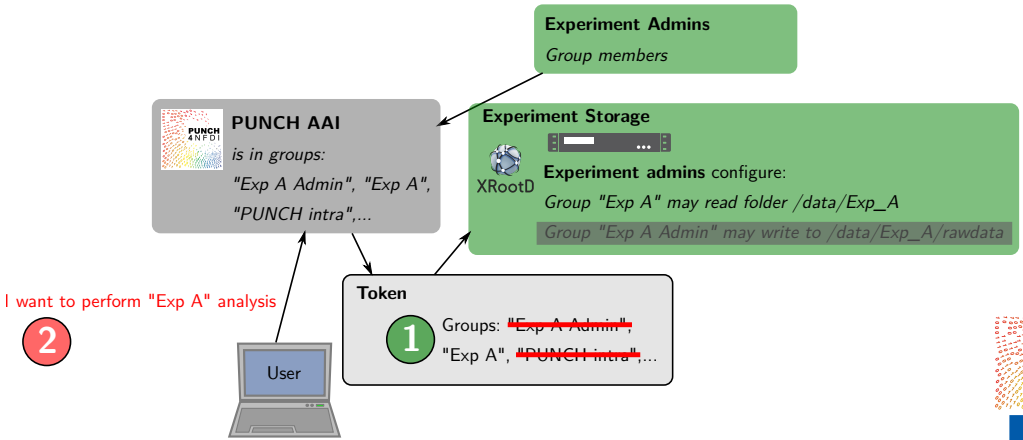
# PUNCH / Helmholtz AAI: Requested extensions

1. embed group information in tokens
   *Allows to authorize decentralized at the compute / storage resources.*
2. reduced permissions in tokens (e. g. not all groups)
   *Allows to work with different roles / reduced permissions.*
3. query (potentially 'third-party') service for granular information and embed this in the token
   *State-of-the-art approach: Granular, path-based authorization.*
   Three realizations:
   1. Embedded inside the AAI (policy engine), e. g. Indigo IAM
   2. Behind the AAI, queried by Unity (REST API)
   3. In front of the AAI (contacted by users instead of the AAI, Trusted AAI Client) uses Token Exchange, approach chosen e. g. by ILDG community

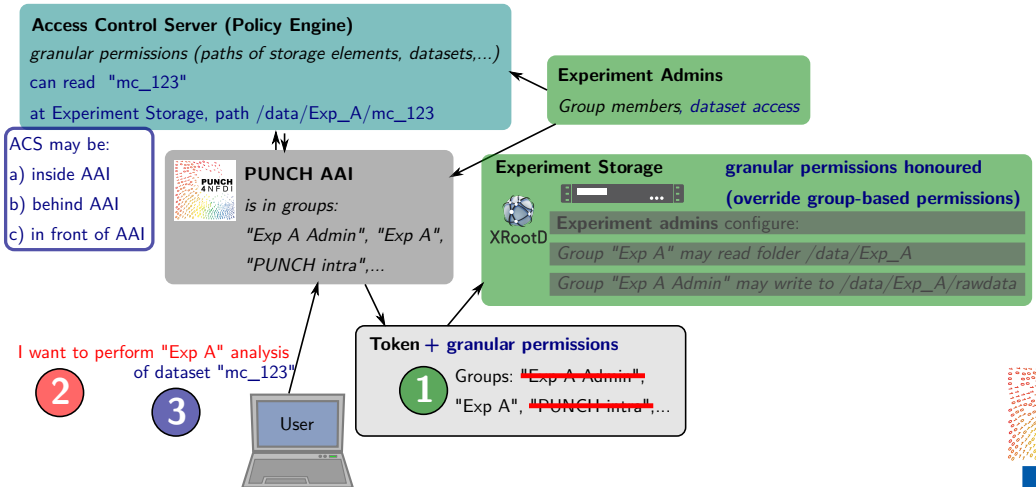
# Visualization of Feature Requests



# Visualization of Feature Requests



# Visualization of Feature Requests



# Status of Requests

- Request 1 ‘Group information in tokens’:
  ✓ Implemented by Unity before we made an official request
  ⇒ Still needs to be tested in full within PUNCH
- Request 2 ‘Claim filtering’ (i. e. not all groups in tokens):
  ⧗ Currently being implemented,
  Unity IAM developers only need about 30 days for actual implementation
- Request 3 ‘Granular permissions’
  ⧗ Description finalized, getting quotation for different possibilities

# Granular Authorization

## Storage

Scopes: `storage.read`, `storage.create`, `storage.modify`, `storage.write`

- Usually scoped for a specific path, i. e. `storage.read:/someexperiment/somefile.dat`
- Inspired by WLCG Common JWT Profiles which are inspired by SciTokens
- Used in the tools in Storage4PUNCH: dCache, XRootD
- May also be used inside Rucio / FTS (now or in the future)

## Compute

Scopes: `compute.read`, `compute.modify`, `compute.create`, `compute.cancel`

- May be scoped to a specific compute resource
- Expected by Compute4PUNCH batch system (HTCondor)
- Inspired by WLCG Common JWT Profiles which are inspired by SciTokens

# Necessary Developments, part 1

**Authorization service** with policy engine required (it must answer questions like 'can user X read from path Y') required.
*Note: Information also needed in Research Product Registry*

## Implementation / Adaption and Operation needed

- 'Off the shelf' policy engines exist, e. g. OpenPolicyAgent
- Integration with PUNCH tools (e. g. also Research Product Registry) and the AAI needed
- Can be worked on after the feature request is accepted

## Note: Indigo IAM

Already embedded. Development work to integrate with the RPR or express more complex policies might be needed.

# Necessary Developments, part 2

AAI is used 'NFDI-wide', control of `storage.read` and similar on global scope unlikely.
*We can likely register* ***`PUNCH-storage.read`*** *and similar.*

## Implementation / Adaption needed

- All tools currently support the WLCG JWT scheme and / or SciTokens
  *Note: SciTokens use `read:/`, `write:/`, `queue:/` & `execute:/`*
- AAI provides more standardized / generic `at+jwt` tokens
- Tools do not handle 'generic' authorization tokens, especially not with different claim names such as `PUNCH-storage.read:/`
- Implementation / adaptation needed for (non-exhaustive list):
  - `scitokens-cpp` library (used by XRootD and HTCondor)
  - dCache (dedicated implementation)
  - Rucio / FTS?

# Necessary Developments, part 2

AAI is used 'NFDI-wide', control of `storage.read` and similar on global scope unlikely.
*We can likely register* ***`PUNCH-storage.read`*** *and similar.*

**Note: Indigo IAM**

No developments needed, as the tools have been developed against that.
Necessary developments for Unity IAM are towards generalizing the tools (not hardcoding WLCG specifics such as claim names and token types) and hence would be helpful to a broader audience.

## Helmholtz AAI / Unity IAM

- Groups can be embedded in tokens since quite a while (basic authentication and authorization with tokens possible for storage).
- Group filtering in development now.
- Granular authorizations require implementations both on AAI and on our end, detailed plan and estimate expected next week
- Scopes required for **compute** missing (requires last feature request).
- Developments necessary to make tools tools more generic for compatibility.
- HIFIS is involved and interested in feature extensions of Helmholtz ID for general purposes as e. g. OpenPolicy import

## General Comments

- Developments on AAI and tools will take significant time (Unity developers are fast, but clear definition, testing, tool evolution take time).
- Making WLCG tools more generic will enable a broader audience (and seems welcome upstream).

Thank you

for your attention!

# Embed group information in tokens

For now, special procedure with `oidc-agent` required (see documentation, simplification foreseen).

## Access Token


```
{
 [...],
 "preferred_username": "o.freyermuth",
 "scope": "openid offline_access profile",
 "eduperson_entitlement": [
  "urn:mace:dir:entitlement:common-lib-terms",
  "urn:geant:dfn.de:nfdi.de:punch:group:PUNCH4NFDI:punch_intra#login.helmholtz.de",
  "urn:geant:h-df.de:group:HDF#login.helmholtz.de",
  "urn:geant:dfn.de:nfdi.de:punch:group:PUNCH4NFDI#login.helmholtz.de",
  "urn:geant:dfn.de:nfdi.de:punch:group:PUNCH4NFDI:elsa_one#login.helmholtz.de"
 ]
}
```

# Reduced permissions in tokens

- Do not expose all groups to used service
- Work with reduced / minimal privileges

## Token Request

```
scope=eduperson_entitlement:...:PUNCH4NFDI:elsa_one#login.helmholtz.de
```

## Access Token

```
{
 [...],
 "preferred_username": "o.freyermuth",
 "scope": "openid offline_access profile
 ↪  eduperson_entitlement:...:PUNCH4NFDI:elsa_one#login.helmholtz.de",
 "eduperson_entitlement": [
  "urn:geant:dfn.de:nfdi.de:punch:group:PUNCH4NFDI:elsa_one#login.helmholtz.de"
 ]
}
```

# Query PUNCH service for granular permissions

- Granular file access permissions require a scope policy system
- Extension in Unity not likely ⇒ external Access Control Server

## Token Request

```
scope=storage.read:/example/subdir/file
```

## Access Token

```
{
 [...],
 "preferred_username": "o.freyermuth",
 "scope": "openid offline_access profile storage.read:/example/subdir/file",
 "storage": {
          [
   "storage.read:/example/subdir/file"
  ]
 }
}
```