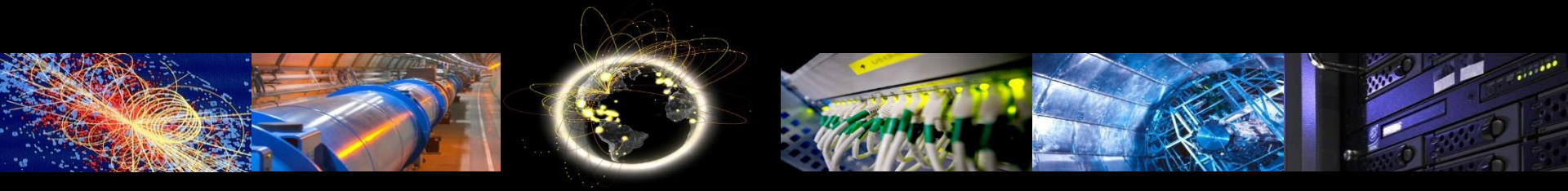
Authored by the WLCG AuthZ WG
Presented by Tom Dack, STFC

WLCG Token Usage and Discovery

vCHEP

May 20th 2021

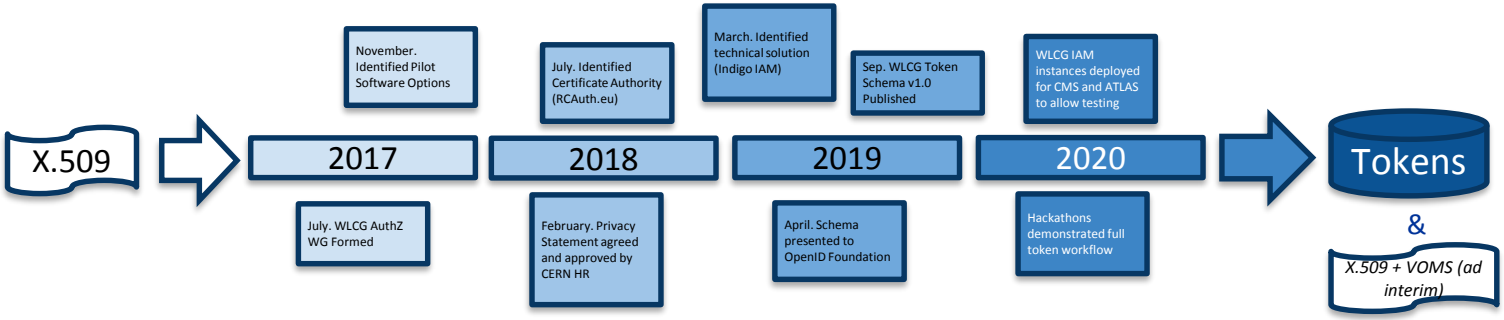


WLCG Authorization (AuthZ) WG

- Membership includes current major users of tokens in High Energy Physics
  - INDIGO IAM
  - EGI Check-in
  - SciTokens
  - dCache
  - ALICE
- Development work of pilot projects supported by:

- Priority to stick to industry and R&E standards wherever possible
- Collaboration with DOMA Working Group - essential for accelerating token support in workflows

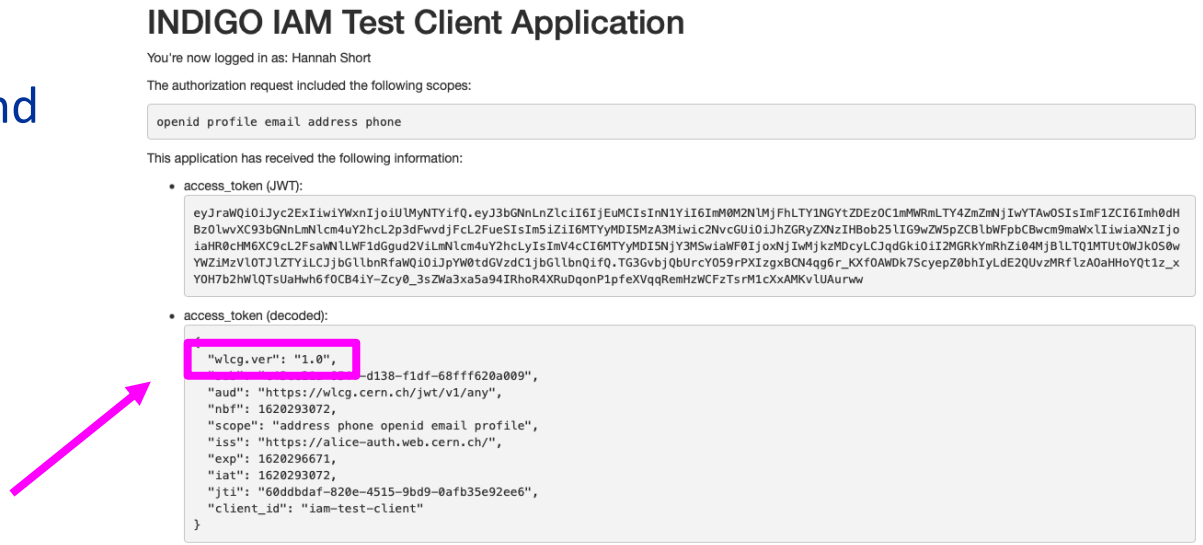
Towards Tokens for WLCG



- Infrastructure
  - Token issuers per VO (Indigo IAM) - *ATLAS and CMS in progress*
    - Operational support and maintenance (CERN IT)
  - Command line solution (Hashicorp Vault) - *Pilot set up*
  - Token support throughout WLCG stack - *Timeline TBC*
- Documentation
  - ✓ Token schema v1.0
  - ✓ Token discoverability specification v1.0

What is a WLCG token?

- JSON Web Token (JWT)
- Distributed over OAuth2.0 Protocol
- Contains identity and authorisation information from issuer (VO)
  - Groups and/or Capabilities
- Follows the WLCG Token Schema (https://zenodo.org/record/3460258)



Example token from the IAM Test Client

Token Discovery

- Many tools will rely on tokens being stored in the local environment
- Token discoverability specification v1.0 published
  https://zenodo.org/record/3937438

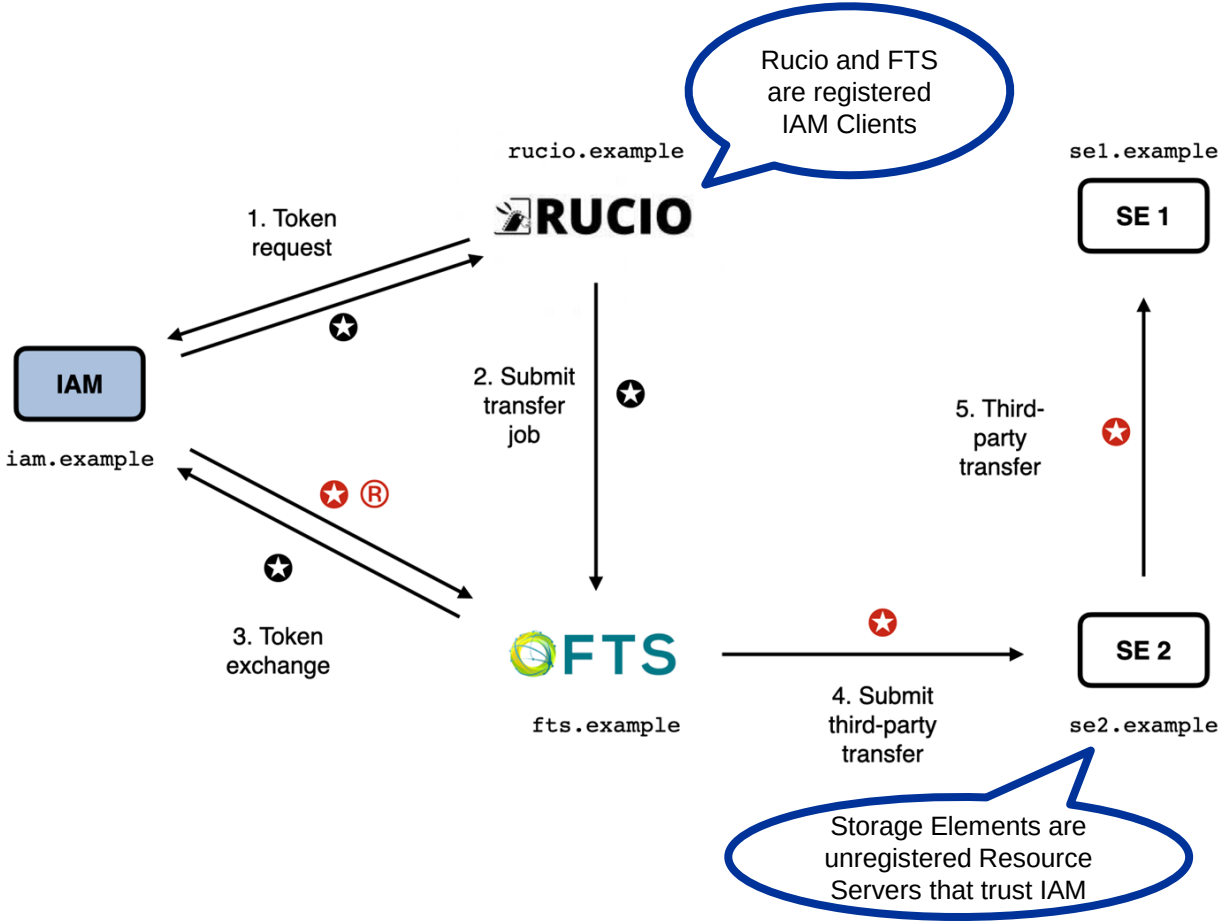
Logic of where to search for (or place) tokens locally

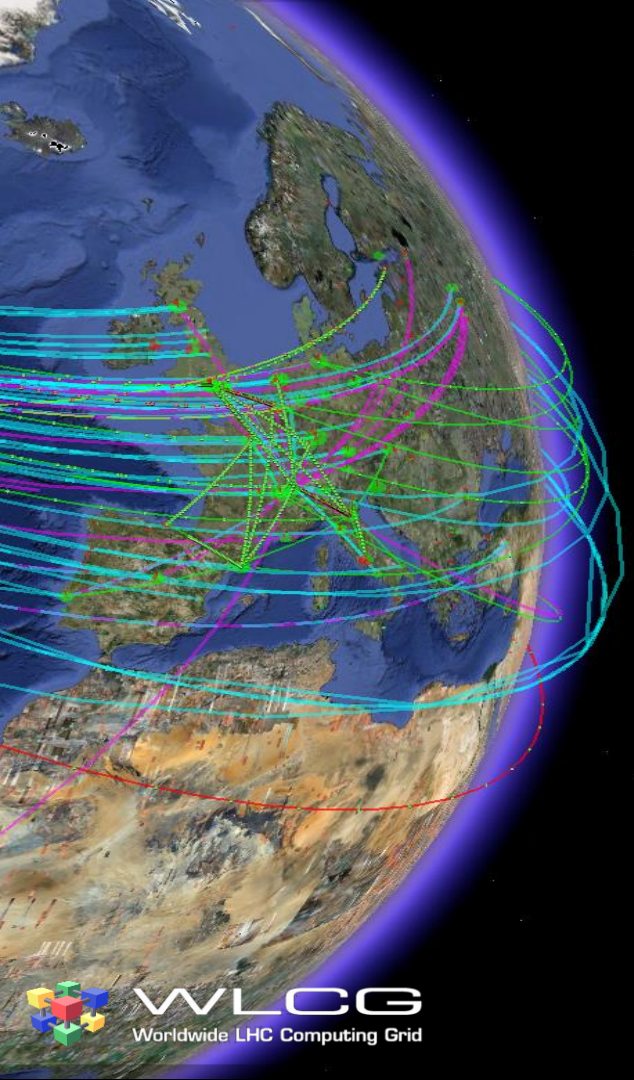
If a tool needs to authenticate with a token and does not have out-of-band WLCG Bearer Token Discovery knowledge on which token to use, the following steps to discover a token MUST be taken in sequence, where $ID below denotes the process's effective user ID:

1. If the **BEARER_TOKEN** environment variable is set, then its value is taken to be the token contents.
2. If the **BEARER_TOKEN_FILE** environment variable is set, then its value is interpreted as a filename. The contents of the specified file are taken to be the token contents.
3. If the **XDG_RUNTIME_DIR** environment variable is set1, then take the token from the contents of $XDG_RUNTIME_DIR/bt_u$ID2.
4. Otherwise, take the token from **/tmp/bt_u$ID**

Rucio-FTS-SEs flow

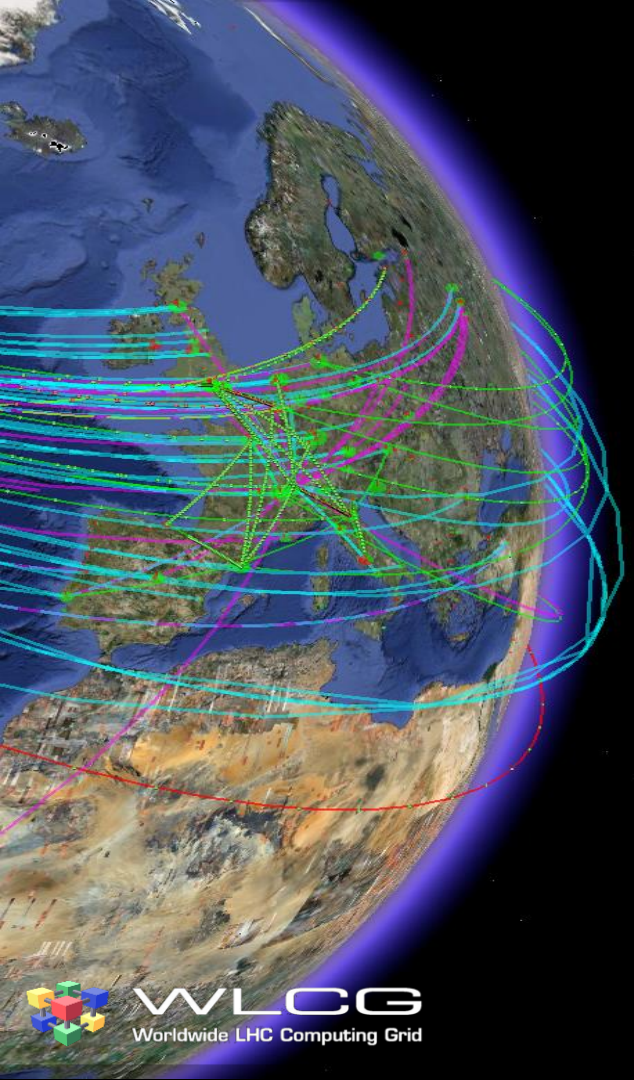
1. Rucio requests token for FTS from IAM
2. Rucio submits job to FTS and includes token
3. FTS exchanges token for one for target third-party
4. Third-party transfer submitted along with new token
5. Token can be reused among instances of third-party





Questions?

WLCG
Worldwide LHC Computing Grid

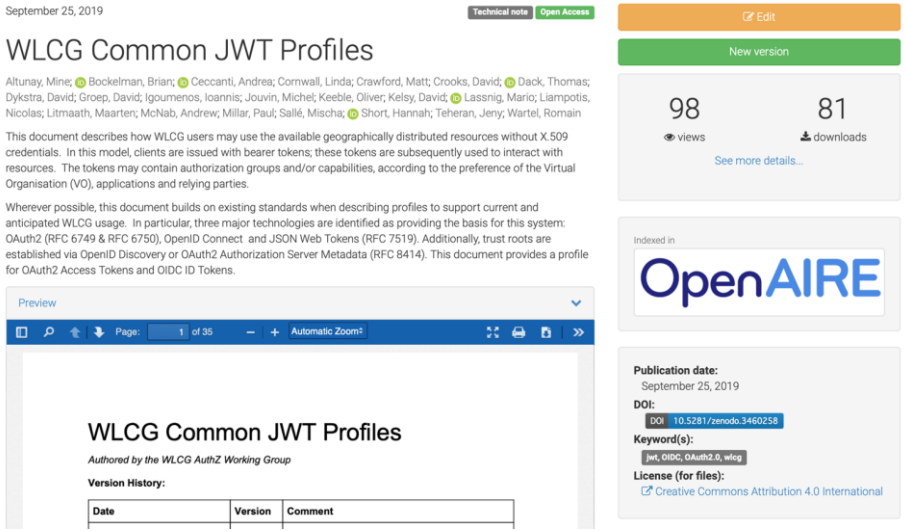


Supplementary Slides

WLCG
Worldwide LHC Computing Grid

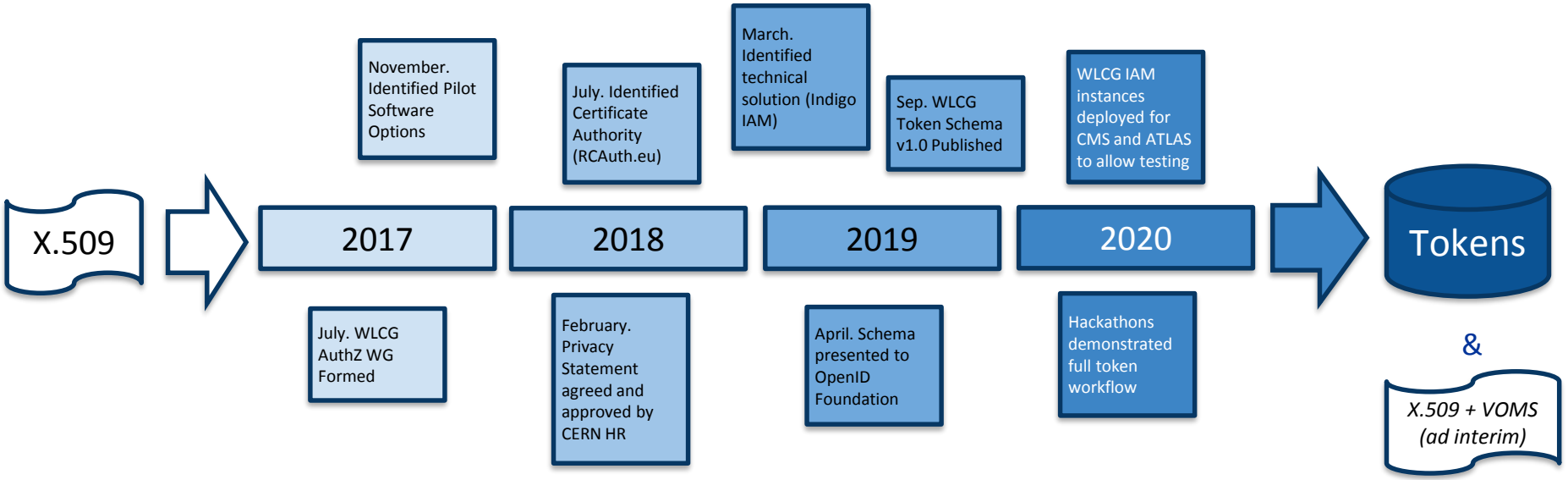
WLCG Schema V1.0

- Published on Zenodo, September 25th 2019
- Allows middleware developers to enable token based authorization to an agreed schema
- Working document at https://github.com/WLCG-AuthZ-WG/common-jwt-profile

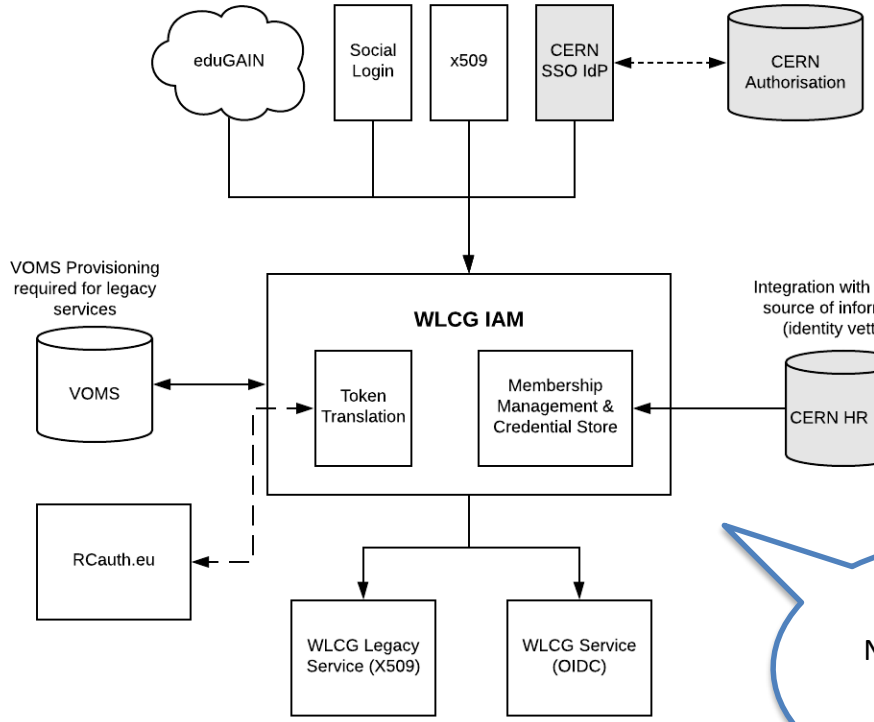


https://zenodo.org/record/3460258

Towards Tokens

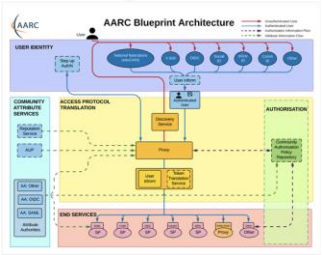


Infrastructure Design



CERN SSO configured as sole Identity Provider, enables identity verification via HR DB (match CERN PersonID)

Follows the AARC Blueprint https://aarc-community.org/architecture/

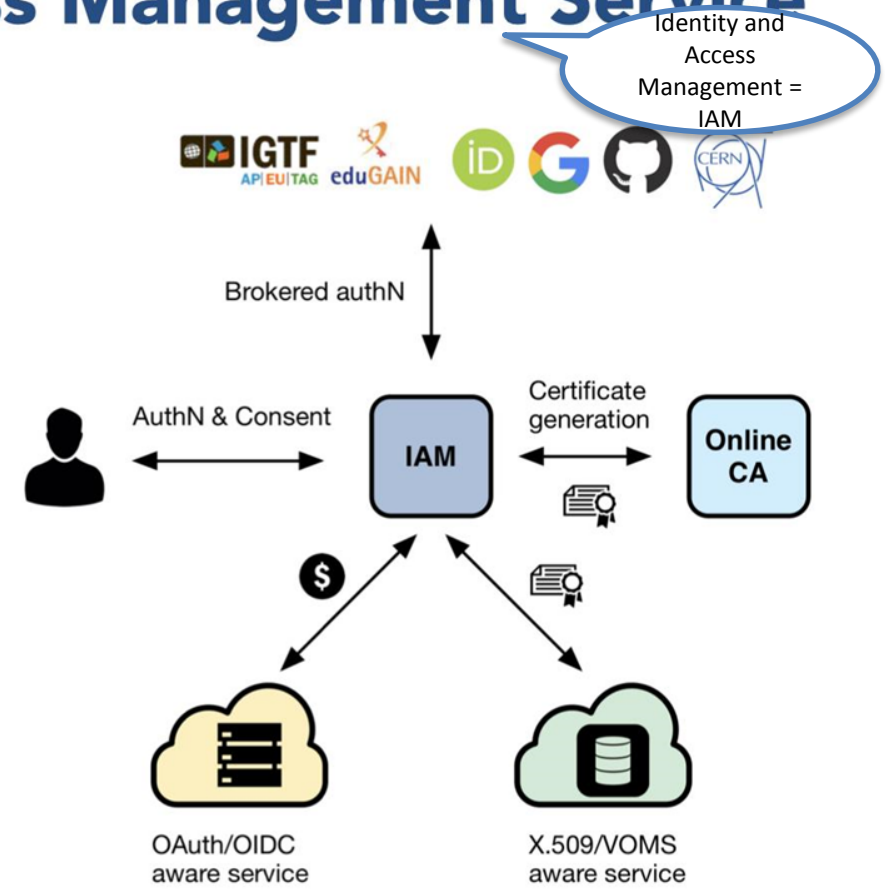


Note: one WLCG AAI per VO

INDIGO Identity and Access Management Service

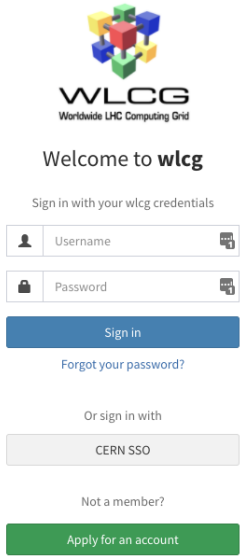
A **VO-scoped** authentication and authorization service that

- 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**

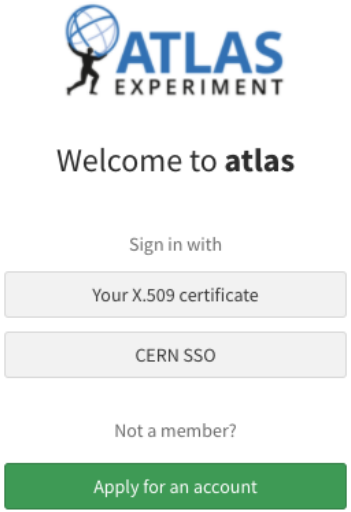


Deployments

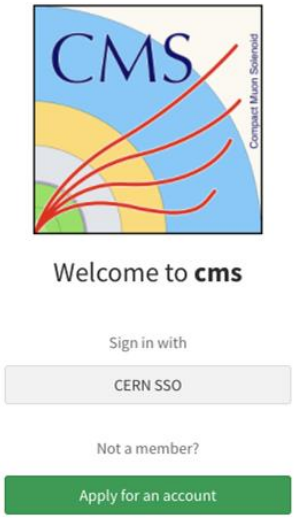
The following token issuers have been deployed. The ATLAS and CMS instances are available for testing and integration, with the expectation that they will become the future production token issuers.



https://wlcg.cloud.cnaf.infn.it



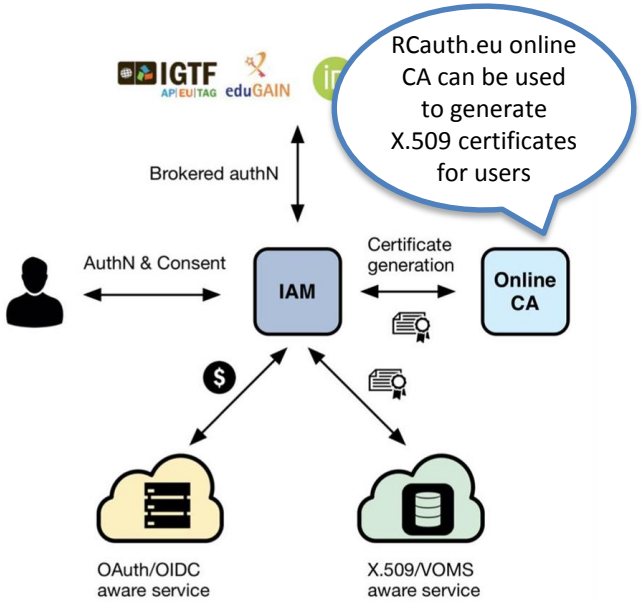
https://atlas-auth.web.cern.ch



https://cms-auth.web.cern.ch

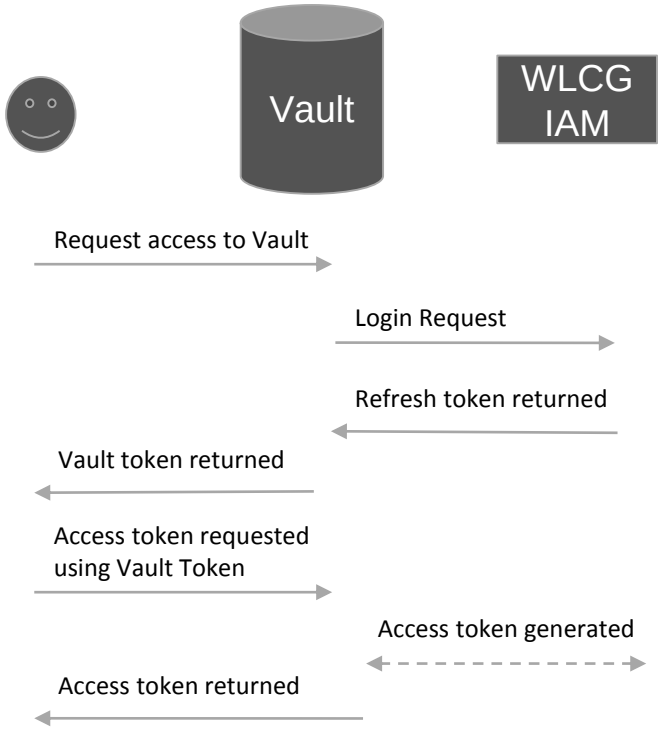
X.509 Backwards Compatibility

- IAM can act as a backwards compatible VOMS server supporting voms-proxy-init etc.
- For users without an end user certificate, RCauth.eu can be used to generate X.509 certificates which are stored in IAM and returned on demand
- Users can be imported from a VOMS server

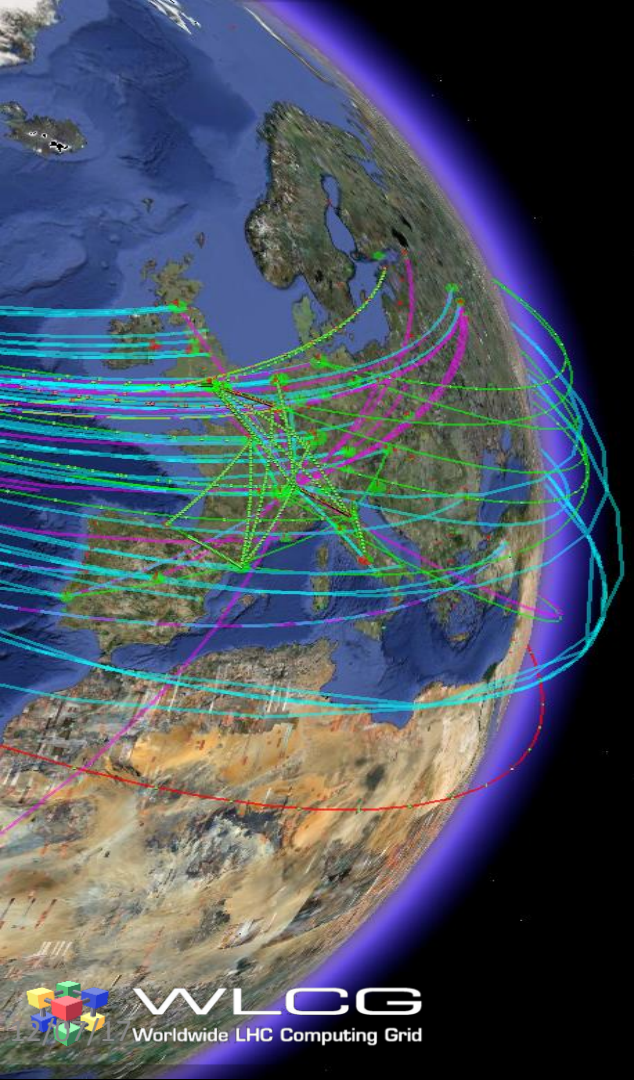


Command Line Tools

- Since many workflows are performed on the command line, we need users to be able to get Tokens in their local environment
  - Must be user friendly
  - Must be secure (i.e. refresh tokens protected)
- Solution identified: **use Hashicorp Vault as the registered client and manager of refresh tokens**



See https://github.com/fermitools/htvault-config



WLCG Token Schema

Token Claims

Common Claims

- sub
- exp
- iss
- acr
- aud
- iat
- nbf
- jti
- eduperson_assurance (REFEDS)
- wlcg.ver (WLCG)
- wlcg.groups (WLCG)

iss+sub used to uniquely identify a user, e.g. for blocking

wlcg prefix added to avoid collisions with other schemas

ID Tokens

- auth_time
- general OIDC Claims

Access Tokens

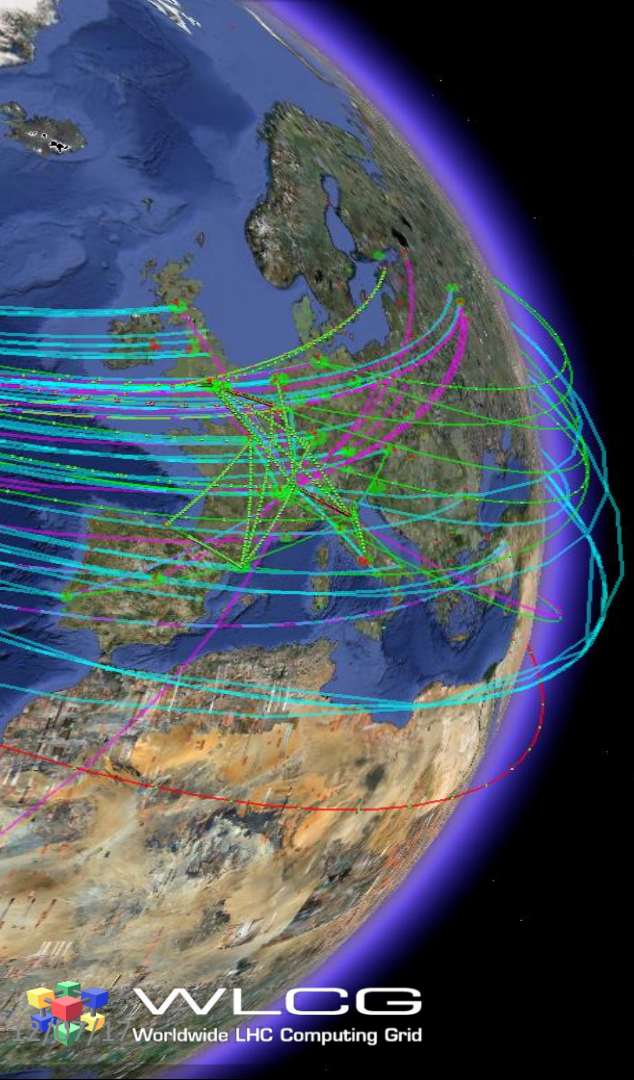
- scope (RFC8693)

Access tokens should include at least scope (capabilities) or group for authorization

Note: Where unspecified, the origin is RFC7519 or OpenID Connect core

Lifetimes

Token Type	Recommended Lifetime	Minimum Lifetime	Maximum Lifetime	Justification
Access Token & ID Token	20 minutes	5 minutes	6 hours	Access token lifetime should be short as there is no revocation mechanism. The granted lifetime has implications for the maximum allowable downtime of the Access Token server.
Refresh Token	10 days	1 day	30 days	Refresh token lifetimes should be kept bounded, but can be longer-lived as they are revocable. Meant to be long-lived enough to be on a “human timescale”.
Issuer Public Key Cache	6 hours	1 hour	1 day	The public key cache lifetime defines the minimum revocation time of the public key. The actual lifetime is the maximum allowable downtime of the public key server
Issuer Public Key	6 months	2 days	12 months	JWT has built-in mechanisms for key rotation; these do not need to live as long as CAs. This may evolve following operational experience, provision should be made for flexible lifetimes.



Authorization

Authorization

- Two models
  - Groups e.g. /atlas/production
  - Capabilities e.g. storage.read/atlas

"Access tokens may convey authorization information as both groups and capabilities. If both group membership and capabilities are asserted, then the resource server should grant the union of all authorizations for the groups and capabilities that it understands." From the WLCG Token Schema

Groups (in wlcg.groups claim)

- Authorization may be based on the **wlcg.groups** claim.
- **wlcg.groups** semantics are equivalent to existing VOMS groups. VOMS roles should be considered as optional (i.e. returned only if requested) **wlcg.groups**
- Requesting the **wlcg.groups** scope returns all default groups

Capabilities (in scope claim)

- Authorization may be based on the **scope** claim.
- Format **$AUTHZ:$PATH** where **$PATH**is mandatory (may be '/' for *)

Scopes proposed for storage and compute

For a given storage resource, the defined authorizations include:

- **storage.read**: Read data. Only applies to "online" resources such as disk (as opposed to "nearline" such as tape where the **stage** authorization should be used in addition).
- **storage.create**: Upload data. This includes renaming files if the destination file does not already exist. This capability includes the creation of directories and subdirectories at the specified path, and the creation of any non-existent directories required to create the path itself (note the server implementation MUST NOT automatically create directories for a client). This authorization DOES NOT permit overwriting or deletion of stored data. The driving use case for a separate `storage.create` scope is to enable stage-out of data from jobs on a worker node.
- **storage.modify**: Change data. This includes renaming files, creating new files, and writing data. This permission includes overwriting or replacing stored data in addition to deleting or truncating data. This is a strict superset of `storage.create`.
- **storage.stage**: Read the data, potentially causing data to be staged from a nearline resource to an online resource. This is a superset of `storage.read`.

For a given computing resource, the defined authorization activities include:

- **compute.read:** "Read" or query information about job status and attributes.
- **compute.modify:** Modify or change the attributes of an existing job.
- **compute.create:** Create or submit a new job at the computing resource.
- **compute.cancel:** Delete a job from the computing resource, potentially terminating a running job.

Scope Based Attribute Selection

- We propose to use scopes to implement an attribute selection mechanism for **both groups and capabilities** following the approach outlined in the OpenID Connect standard:
  - https://openid.net/specs/openid-connect-core-1_0.html#ScopeClaims
- Authorizations are requested using scopes and returned by the token issuer if the client and user are entitled

Scope Based Attribute Selection

Scope Request	Claim Result
`scope=storage.read:/home/joe`	`"scope": "storage.read:/home/joe"`
`scope=storage.read:/home/joe storage.read:/home/bob`	`"scope": "storage.read:/home/joe storage.read:/home/bob"`
`scope=storage.create:/ storage.read:/home/bob`	`"scope": "storage.create:/ storage.read:/home/bob"`

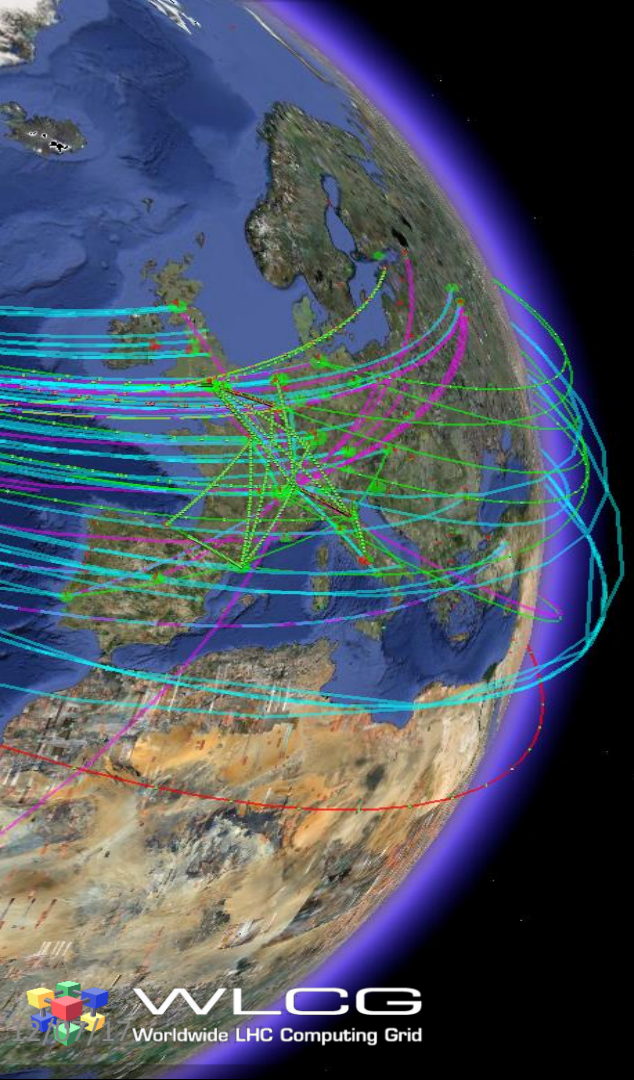
Capability requests are matched exactly

Scope Request	Claim Result
`scope=wlcg.groups`	`"wlcg.groups": ["/cms"]`
`scope=wlcg.groups:/cms/uscms wlcg.groups:/cms/ALARM`	`"wlcg.groups": ["/cms/uscms","/cms/ALARM", "/cms"]`
`scope=wlcg.groups:/cms/uscms wlcg.groups:/cms/ALARM wlcg.groups`	`"wlcg.groups": ["/cms/uscms","/cms/ALARM", "/cms"]`
`scope=wlcg.groups wlcg.groups:/cms/uscms wlcg.groups:/cms/ALARM`	`"wlcg.groups": ["/cms", "/cms/uscms","/cms/ALARM"]`
`scope=wlcg.groups:/cms wlcg.groups:/cms/uscms wlcg.groups:/cms/ALARM`	`"wlcg.groups": ["/cms", "/cms/uscms","/cms/ALARM"]`

/cms is a default group and always returned

Capability Sets

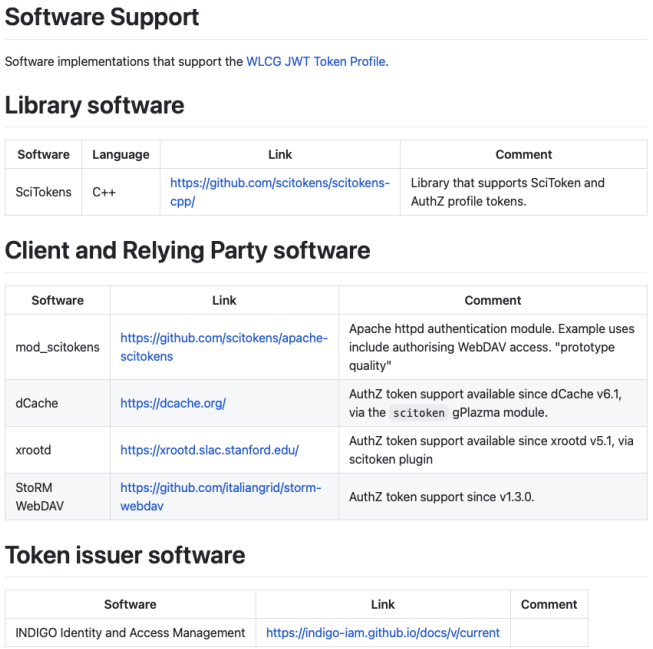
- In some use cases (e.g. Vault) a client may not know exactly which capabilities will be required by downstream services
- A capability set can be requested; if granted, multiple capabilities will be returned in the token
- Work ongoing at https://github.com/WLCG-AuthZ-WG/common-jwt-profile/pull/10



Next Steps

Schema Adoption

- A catalogue of software that supports the WLCG JWT Schema is being compiled at https://github.com/WLCG-AuthZ-WG/software-support
- Aware that storage components are progressing faster than compute



Software Support

Software implementations that support the WLCG JWT Token Profile.

Library software

Software	Language	Link	Comment
SciTokens	C++	https://github.com/scitokens/scitokens-cpp/	Library that supports SciToken and AuthZ profile tokens.

Client and Relying Party software

Software	Link	Comment
mod_scitokens	https://github.com/scitokens/apache-scitokens	Apache httpd authentication module. Example uses include authorising WebDAV access. "prototype quality"
dCache	https://dcache.org/	AuthZ token support available since dCache v6.1, via the `scitoken` gPlazma module.
xrootd	https://xrootd.slac.stanford.edu/	AuthZ token support available since xrootd v5.1, via scitoken plugin
StoRM WebDAV	https://github.com/italiangrid/storm-webdav	AuthZ token support since v1.3.0.

Token issuer software

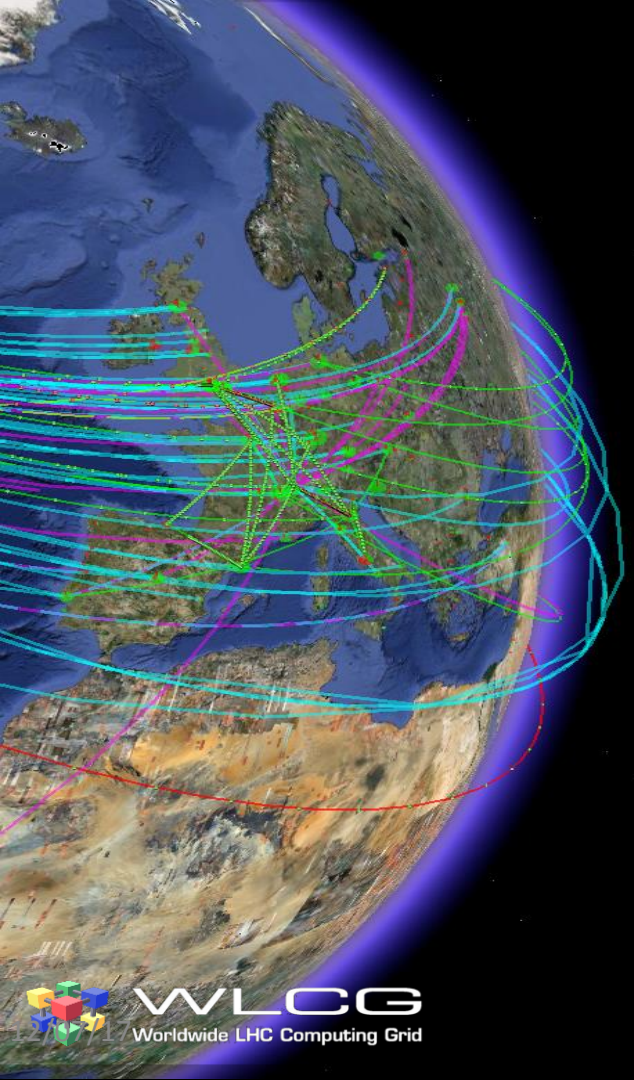
Software	Link	Comment
INDIGO Identity and Access Management	https://indigo-iam.github.io/docs/v/current	

Interoperability

- Participating in AEGIS community to facilitate interoperability between infrastructures e.g. EOSC https://aarc-project.eu/about/aegis/

Timeline

- Timeline under discussion https://twiki.cern.ch/twiki/bin/view/LCG/WLCGTokensGlobusWG
- Summary at https://indico.cern.ch/event/876787/#6-globus-retirement-timeline
- Key points (TBC)
  - Q2 2021 production IAMs available for VOs
  - Q3 2021 pilot jobs may be performed with tokens
  - Q4 2021 VOMS-Admin retired



Questions?