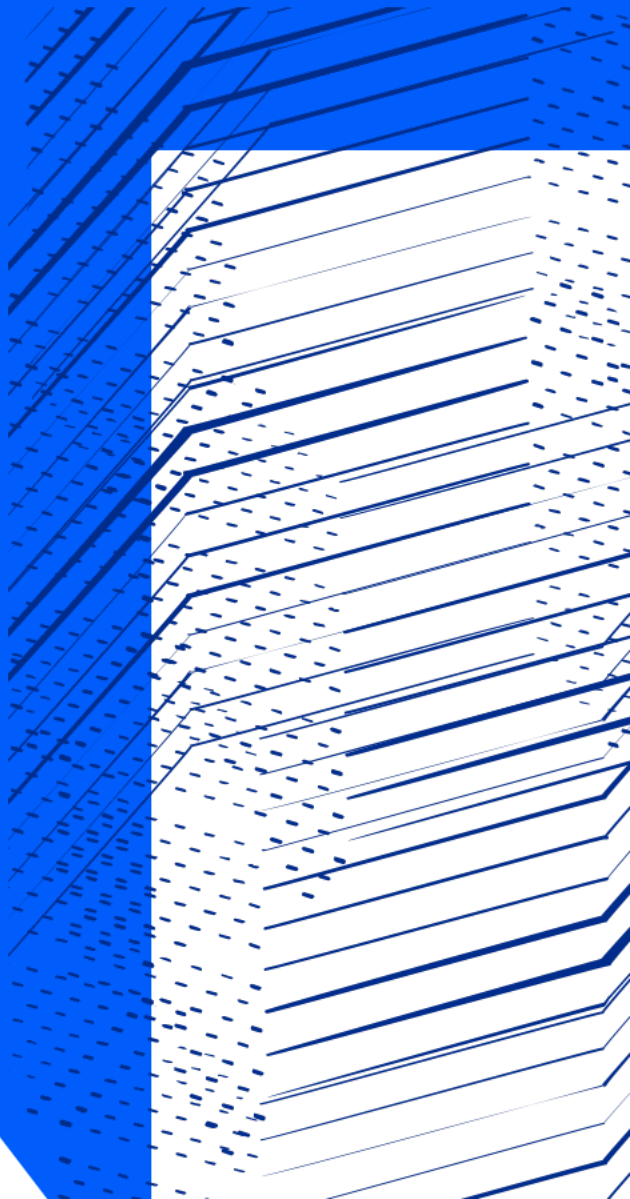
Authentication and Authorisation for the WLCG

An introduction to the key concepts, and how they fit within the WLCG Space



Who am I?

I currently wear multiple Authentication and Authorisation hats…

- Service Manager for the Identity and Access Management service for the U.K. IRIS Collaboration
- Current Authorisation Working Group Chair for the WLCG
- Scrum master for the SKA agile team working on Authentication and Authorisation Infrastructure (AAI)

Working to ensure that all these communities (and others!) can interoperate

Questions? Feel free to contact at:
thomas.dack@stfc.ac.uk

Science and Technology Facilities Council

Agenda

1 Introduction to AuthN & AuthZ

…and why does it matter anyway

2 AuthN & AuthZ For the WLCG

3 AuthN & AuthZ: Certificates & VOMS

4 AuthN & AuthZ: Tokens

5 Towards Tokens for WLCG

Science and Technology Facilities Council

Image © STFC Alan Ford

Take Aways

- What is the difference between Authentication and Authorisation
- An understanding of why this is important for research
- A basic understanding of the existing Certificate and VOMs infrastructure
- A more in-depth understanding of the OAuth and OIDC protocols
- A brief overview of the planned token transition for WLCG

Science and
Technology
Facilities Council

Introduction to Authentication & Authorisation

…and why does it matter anyway

Authentication and Authorisation

Or, Not letting everyone in

- Letting everyone access everything is often *a bad idea*
- Though this is not always true – the level of access control required involves considering the risks
- In the contest of research, we probably don't want anyone and everyone online being able to get in…

Science and Technology Facilities Council

Authentication and Authorisation

Controlling access depends on:

- Verifying a user's access to an account or identity – **authentication (AuthN)**
- Knowing that a user is allowed to do what they want to do – **authorisation (AuthZ)**

These processes are usually combined, as key processes in a community's **Authentication and Authorisation Infrastructure - AAI**

SIDE NOTE:

- **AuthoriZation** - 🇺🇸 & 🇨🇦
- **AuthoriSation** - 🇬🇧 & the rest of the English-speaking world

I will have typed **AuthoriSe**, except in occurrences where it is a **named term** or **attribute**

… also I use the Americanised shorthand, **AuthZ**

Science and Technology Facilities Council

Authentication and Authorisation - Then

Traditionally a user would need to Authenticate separately to every new service or account they wanted to use

- Lots of accounts, scattered through every site a user has ever signed up for
- This results in *many* username and password pairs, which in turn leads to bad security practices
  - Credentials are reused for “simplicity”, or simply forgotten and lost

Authentication and Authorisation - Now

In more modern applications, the process is evolving

- Increased importance of account credibility
  - Processes and systems in place to show that an associated identity is verified
- Increased adoption of single sign-on mechanisms, using a unified identity
  - Log-in with Social IDs, such as Facebook, Google, ORCID
- Can grant authorised access based on an identity's attributes
  - Home institute, email address, etc

What makes AAI important for research?

A couple of key concepts underpin the use of identity in research:

- Confidentiality
- Traceability
- Suspension
- Attribution

What makes this important for research?

A couple of key concepts underpin this:

- Confidentiality
- Traceability
- Suspension
- Attribution

Confidentiality

- Whilst final research outputs are public, maintaining confidentiality before this point is important
- Particularly important for fields handling personal and medical data

What makes this important for research?

A couple of key concepts underpin this:

- Confidentiality
- Traceability
- Suspension
- Attribution

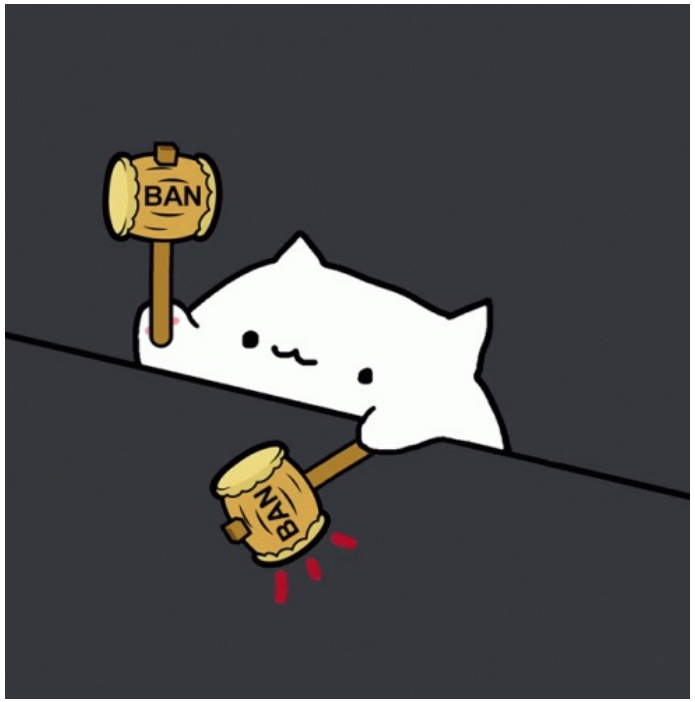
Traceability

- If something goes wrong – accidentally or maliciously – service owners need to be able to trace where this happened
- Knowing which account caused an issues is important for both support or suspension

What makes this important for research?

A couple of key concepts underpin this:

- Confidentiality
- Traceability
- Suspension
- Attribution



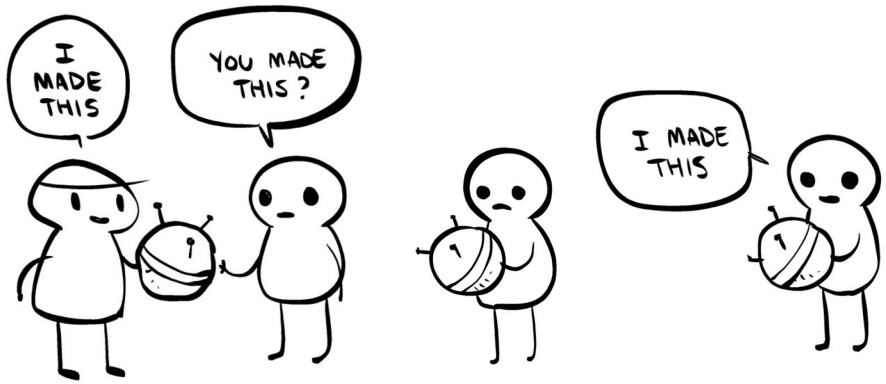
Suspension

- Processes to suspend an individual user account in case of compromise or malicious activity
- Avoids downtime caused by stopping an entire service or resource, by isolating the problematic identity

What makes this important for research?

A couple of key concepts underpin this:

- Confidentiality
- Traceability
- Suspension
- Attribution



Tumblr – The Internet by Nedroid / January 30th, 2013

Attribution

- A single central identity provides a mechanism through which research can be attributed
- ORCID is an example of this – life-long identifiers for researchers, which can be attached to publications, grant requests, etc
- Identity changes can present problems – name changes from marriage, gender transition, etc

Science and Technology Facilities Council

AuthN & AuthZ For the WLCG

AuthN & AuthZ for the WLCG

Providing global access to computing resources – not easy!

- Global user community, with many members
- Distributed single infrastructure
- Not guaranteed that users know each other
- Not guaranteed that users will ever meet

Need a system for provisioning access, to be trusted across the grid

- ***Not guaranteed that users know each other***

Who can you trust to know the users, in order to **authenticate** them?

Many options, including:
- The Infrastructure
- The Experimental Group or Research Community
- The Home Organisations
- A trusted Third Party

In most cases, a user's **Home Organisation** may have the most current information – especially if their access is a function of their affiliation.

- ***Not guaranteed that users will ever meet***

But! A user's **Experimental Group or Research Community** may be better placed to tell you…

- Which group the user belongs to
- What roles (permissions) they should have
  - *what are they **authorised** to do*
  - *Are they a user, an admin, a super-user, etc?*
- The status of a user's policy acceptance – e.g. whether they have accepted the latest acceptable usage policy

AuthN & AuthZ for the WLCG

Need to bring these concepts together in order to control access to the grid

We'll look at an overview of how this is done for the WLCG, looking at both the current system and the intended migration infrastructure:

Current:	Future:
Certificates & VOMS	Tokens & IAM

Science and
Technology
Facilities Council

AuthN & AuthZ: Certificates & VOMS

X.509 – a Recap

A Certificate is…

- A digital identity, representing an entity
  - could be a service/website, a machine, or a human individual
- Signed by a Certificate Authority (CA)
  - Self-signed certificates do exist but are not useful for authentication purposes!
  - Signed by taking a hash of the certificate, which is then encrypted with the CA's private key
- Long lived – typically a grid certificate will last a year
- User keeps an accompanying private key and password

VOMS - Virtual Organization Membership Service

VOMS is the certificate-based AAI for the WLCG

- A central attribute authority, and central repository for VO user information
- Enables sorting of users into group hierarchies, storing roles and other attributes
- This is used to issue trusted attribute certificates which can then be used in the Grid environment for authorisation purposes.

VOMS - Virtual Organization Membership Service

The typical VOMS user flow is as follows

- The user registers at the CERN User Office and with their experiment secretariat, and the user is entered into the CERN HR database.
- The user gets a certificate from any eligible Certificate Authority
- The user registers with VOMS Admin for their VO and presents their certificate to authenticate. Their email address is matched with their record in the CERN HR database to confirm that their identity has been verified. The VO manager approves the registration. From that time on, the HR ID is used as the unique ID in the VOMS DB, allowing the e-mail address to be changed independently at either end.
- VOMS attributes, such as groups and roles, are maintained by VO Managers.
- The user can submit jobs to the grid or access data on the grid by saving their User Certificate locally, running voms-proxy-init to generate the VOMS proxy, and running the relevant grid command; their proxy will get delegated to grid services as needed.

Proxy Certificates?

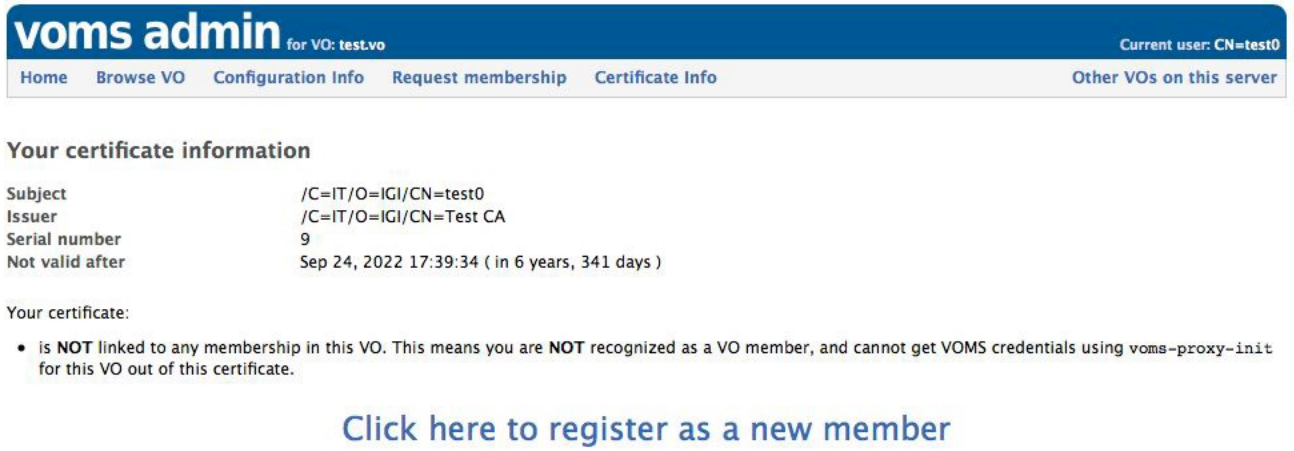
A Proxy Certificate is generated and signed using a user's certificate

- As it is signed by the user, the proxy certificate acts as a proof of identity – can be used for **authentication**

A VOMS Proxy adds extra, VO specific, information to the proxy – such as roles

- This adds **authorisation** information to the proxy

Proxy certificates are short lived, and can be used to in turn sign further proxies – therefore allowing for delegated access on behalf of the user



Science and Technology Facilities Council

AuthN & AuthZ: Tokens

…Tokens?

There are a few components in a token flow. Key to this, a token is…

- a JSON Web Token
  - “JWTs are an open, industry standard RFC 7519 method for representing claims securely between two parties.” https://jwt.io/introduction/
  - Tokens are encoded strings of data, issued by an issuer with which the client (receiver) has a trusted relationship.
- In this context, JWTs are used to communicate authentication and authorization information using the **OAuth 2.0** and **OIDC** protocols

OAuth 2.0	OIDC
Open **Auth**orization	Open **ID C**onnect

Oauth 2.0 – the Access Token

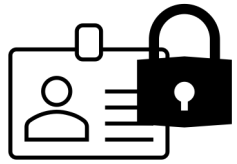
- An open standard for access delegation, and most often used for allowing users to grant a website access to their information on another website, without giving them their password
  - Examples include signing into a third-party website using your Google, GitHub, or Orcid account
- An access token is provided to the third party, from whichever service is acting as the Authorization Server, which it can then use to retrieve a protected resource

Science and Technology Facilities Council

Want to know more about OAuth after the lecture?

https://oauth.net/

Oauth 2.0 – Terminology



- **Protected Resource**
  - The identity or data which is to be shared
  - *eg: your email address, or the ability to post to your Twitter*
- **Resource Owner**
  - The user who owns the **Protected Resource**
  - *eg: you!*

- **Client**
  - The application that wants access on behalf of the **Resource Owner**
  - *eg: the website you want to register with using your Google identity*

- **Authorisation Server**
  - The application which knows the **Resource Owner**, and where the **Resource Owner** already has an account
  - *eg: The Google, Twitters, Orcids of the world*

- **Resource Server**
  - Where **the Protected Resource** lives, and what the **Client** wants to use
  - *eg: the API from which the **Client** can access the **Protected Resource***

UKRI Science and Technology Facilities Council

Oauth 2.0 – Terminology

- **Protected Resource**
  - The identity or data which is to be shared
  - *eg: your email add*
- **Resource Owner**
  - The user who own
  - *eg: you!*
- **Client**
  - The application tha
  - *eg: the website yo*
- **Authorisation Serv**
  - The application wh has an account
  - *eg: The Google, Twit*
- **Resource Server**
  - Where **the Protec**
  - *eg: the API from w*

A quick aside…

OAuth 2.0 defines two types of clients:

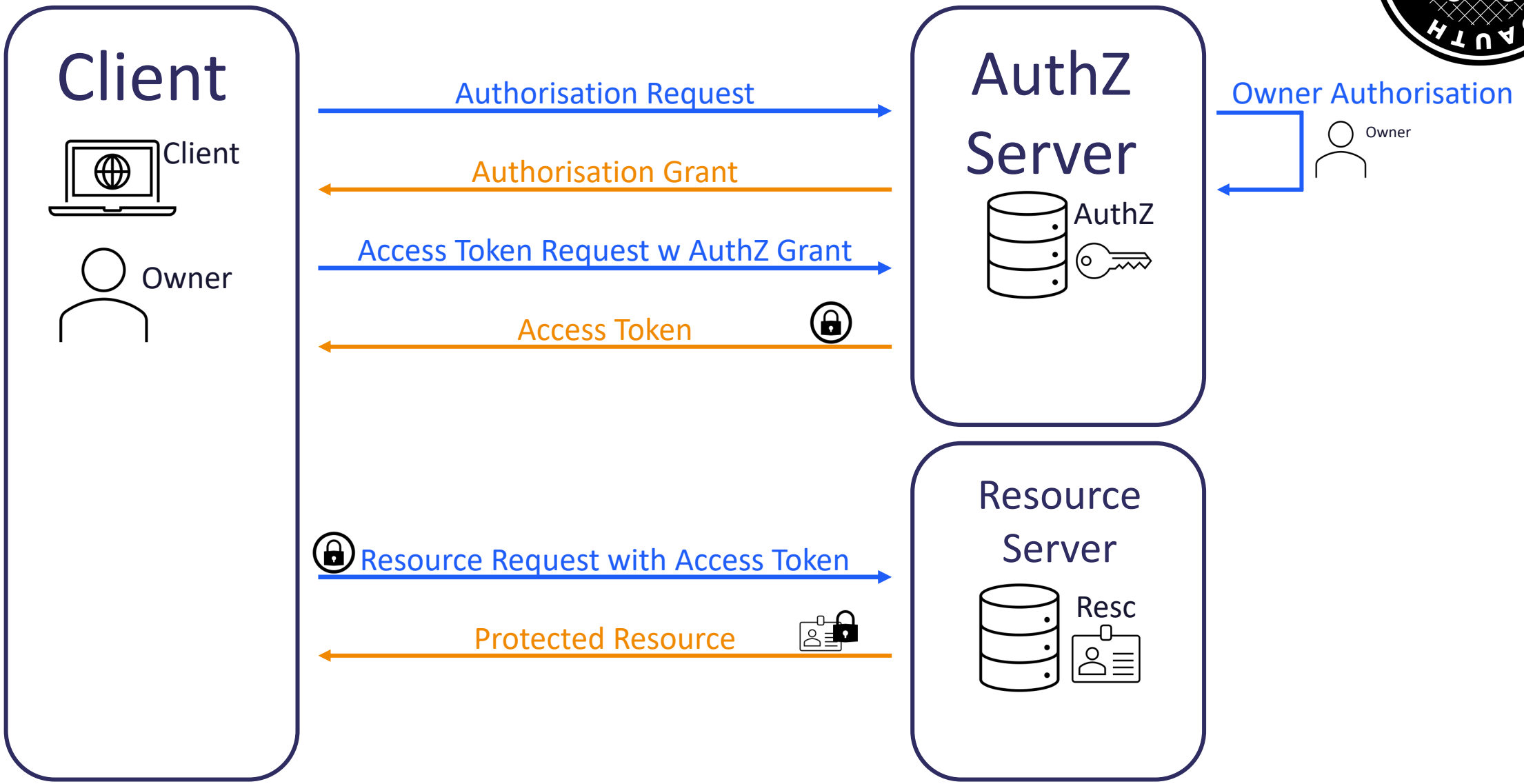
Confidential and **Public Clients**

- Confidential applications can hold credentials with which they use to authenticate themselves to the AuthZ Server in a secure way. They require a trusted backend server to store the secret(s).
- *eg – a web application with a secure backend*

- Public clients **cannot** hold credentials securely.
- *eg – a native desktop or mobile application, or a JavaScript-based client-side web application (single-page app)*

Science and Technology Facilities Council

OAuth 2.0 – How things work



OAuth 2.0 – Authorisation Grants

- The OAuth 2.0 framework specifies several different methods through which a **Client** can verify itself to the **Authorisation Server** - these are known as Authorisation Grants
  - Authorisation Code
  - PKCE
  - Client Credentials
  - Device Code
  - Refresh Token

Key Authorisation Grant Concepts

- Redirect URL – a URL at the client which the AuthZ server will deliver an issued token to
- ClientID – the “username” of the client
- ClientSecret – the “password” of the client

Science and Technology Facilities Council

OAuth 2.0 – Authorisation Grants

- The OAuth 2.0 framework specifies several different methods through which a **Client** can verify itself to the **Authorisation Server** - these are known as Authorisation Grants
  - Authorisation Code
  - PKCE
  - Client Credentials
  - Device Code
  - Refresh Token

Authorisation Code

- Used by both confidential and public clients
- An authorisation code is exchanged for an access token
- When the user returns to the client via the Redirect URL, the Authorisation Code is extracted from the URL and used to obtain the Access Token

UKRI Science and Technology Facilities Council

OAuth 2.0 – Authorisation Grants

- The OAuth 2.0 framework specifies several different methods through which a **Client** can verify itself to the **Authorisation Server** - these are known as Au
  - Authorisa
  - PKCE
  - Client Cr
  - Device C
  - Refresh Token

```
https://example.com/oauth/auth?
    response_type=code
    &scope=EXAMPLE_REQUESTED_SCOPES
    &client_id=EXAMPLE_CLIENTID
    &redirect_uri=EXAMPLE_REDIRECT_URI
```

blic clients

- An authorisation code is exchanged for an access token
- When the user returns to the client via the Redirect URL, the Authorisation Code is extracted from the URL and used to obtain the Access Token

Science and Technology Facilities Council

OAuth 2.0 – Authorisation Grants

- The OAuth 2.0 framework specifies several different methods through which a **Client** can verify itself to the **Authorisation Server** - these are known as Authorisation Grants
  - Authorisation Code
  - Implicit
  - Resource Owner Password Credentials
  - Client Credentials
  - Device Code
  - Refresh Token

```
curl --header "Authorization: Basic EXAMPLE_SECRET"
--data "grant_type=authorization_code&code=EXAMPLE_CODE"
--request POST https://example.com/oauth/token
```

- Used by both confidential and public clients
- An authorisation code is exchanged for an access token
- When the user returns to the client via the Redirect URL, the Authorisation Code is extracted from the URL and used to obtain the Access Token

Science and Technology Facilities Council

OAuth 2.0 – Authorisation Grants

- The OAuth 2.0 framework specifies several different methods through which a **Client** can verify itself to the **Authorisation Server** - these are known as Authorisation Grants
  - Authorisation Code
  - PKCE
  - Client Credentials
  - Device Code
  - Refresh Token

PKCE: *Proof Key for Code Exchange*

- An extension to Authorisation Code, which aims to prevent Cross Site Request Forgery (CSRF) and authorisation code injection attacks
- Not a replacement for a ClientSecret, and therefore does not enable treating a Public Client as Confidential

Science and Technology Facilities Council

OAuth 2.0 – Authorisation Grants

- The OAuth 2.0 framework specifies several different methods through which a **Client** can verify itself to the **Authorisation Server** - these are known as Authorisation Grants
  - Authorisation Code
  - PKCE
  - Client Credentials
  - Device Code
  - Refresh Token

Client Credentials

- Must ONLY be used by Confidential Clients
- The Client authenticates itself directly with the Authorization Server, for example using it's ClientID and ClientSecret pair or with an public/private key pair
- As Client Authentication is used as the Authorisation grant, no further AuthZ is required
- Often used for a client to obtain information about itself

UKRI Science and Technology Facilities Council

OAuth 2.0 – Authorisation Grants

- The OAuth 2.0 framework specifies several different methods through

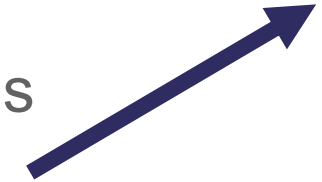
```
curl --request POST \
 --url 'https://EXAMPLE/oauth/token' \
 --header 'content-type: application/x-www-form-urlencoded' \
 --data grant_type=client_credentials \
 --data client_id=EXAMPLE_CLIENT_ID \
 --data client_secret=EXAMPLE_CLIENT_SECRET \
 --data audience=EXAMPLE_AUDIENCE
```

ClientSecret pair or with an public/private key pair

- As Client Authentication is used as the Authorisation grant, no further AuthZ is required
- Often used for a client to obtain information about itself

OAuth 2.0 – Authorisation Grants

- The OAuth 2.0 framework specifies several different methods through which a **Client** can verify itself to the **Authorisation Server** - these are known as Authorisation Grants
  - Authorisation Code
  - PKCE
  - Client Credentials
  - Device Code
  - Refresh Token



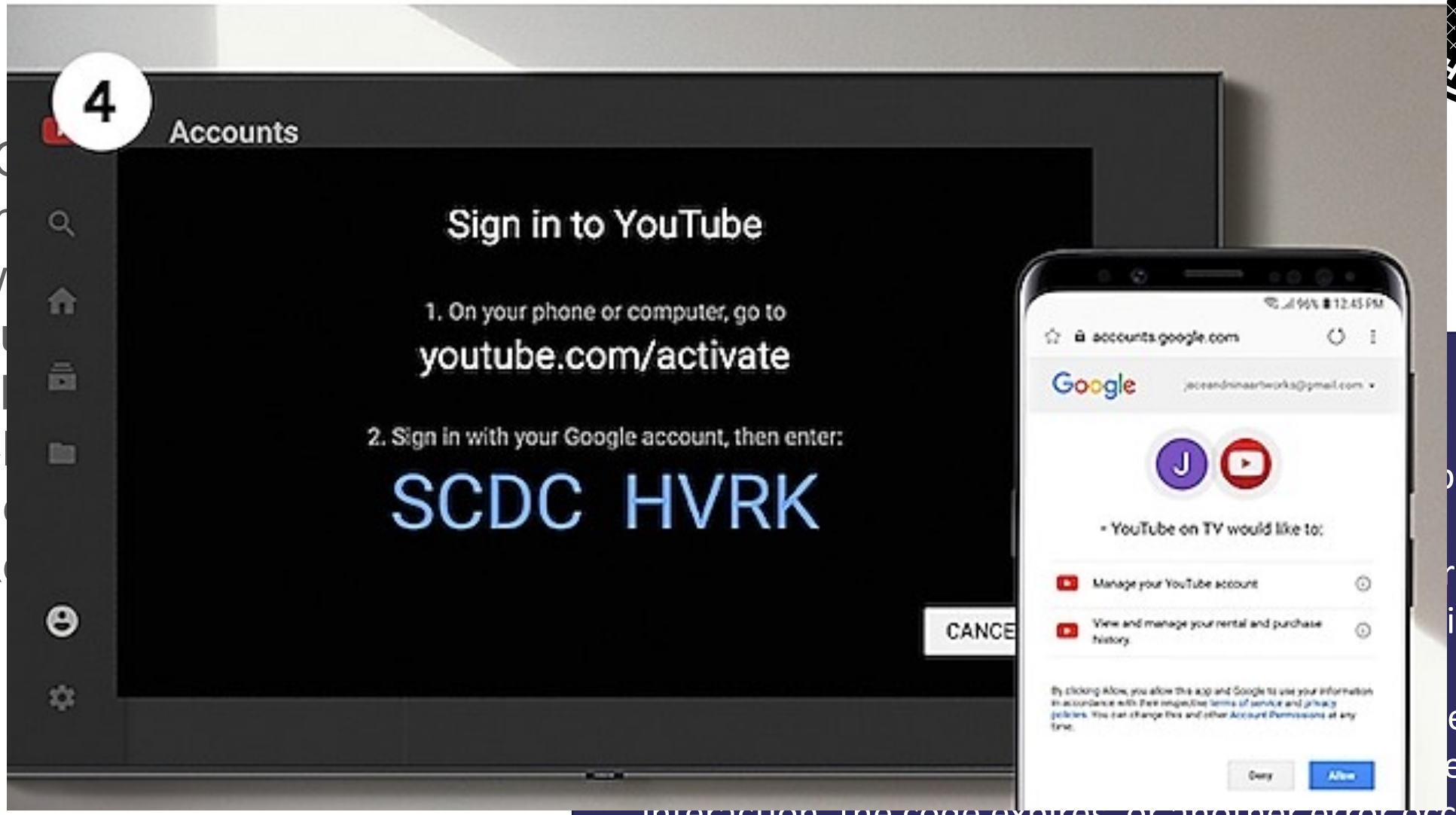
Device Code

- This flow is intended for use by browserless or input constrained clients – such as command line
- The user is directed to visit a URL at the AuthZ Server in a separate browser, along with a user code to identify the device
- The original device continuously polls the AuthZ server with the generated device code until the user completes the interaction, the code expires, or another error occurs

UKRI Science and Technology Facilities Council

OAuth 2.0 – Authorisation Grants

- The G
  which
  know
  - Au
  - Pl
  - Cl
  - De
  - Re



ut

rver in a
ify the

erver with
es the

interaction, the code expires, or another error occurs

Technology Facilities Council

OAuth 2.0 – Authorisation Grants

- The OAuth 2.0 framework specifies several different methods through which a **Client** can verify itself to the **Authorisation Server** - these are known as Authorisation Grants
  - Authorisation Code
  - PKCE
  - Client Credentials
  - Device Code
  - Refresh Token

Refresh Token

- This is a mechanism for allowing a client to get a new Access Token after an initial one has expired, without further user interaction
- Not a replacement for a ClientSecret, and therefore does not enable treating a Public Client as Confidential

UKRI Science and Technology Facilities Council

OAuth 2.0 – Authorisation Grants

To get a Refresh token, the client needs to request the scope:

```
    scope=offline_access&

curl --request POST \
 --url 'https://{yourDomain}/oauth/token' \
 --header 'authorization: Basic {yourApplicationCredentials}' \
 --header 'content-type: application/x-www-form-urlencoded' \
 --data grant_type=refresh_token \
 --data 'client_id={yourClientId}' \
 --data 'refresh_token={yourRefreshToken}
```

therefore does not enable treating a Public Client as Confidential

Science and Technology Facilities Council

Oauth 2.0 – the Access Token

- OAuth is an **Authorisation** flow – does not have a method for user **Authentication**

- Enter…

OpenID Connect: OIDC

Science and Technology Facilities Council

OIDC: The ID Token

- **OpenID Connect (OIDC)** is an identity layer on top of the base OAuth 2.0 protocol
- OIDC provides the ability for app and web developers to authenticate users without the need to directly store credential sets
- OIDC enables the Authorisation Server to act as an **Identity Provider (IdP)**. An OAuth 2.0 Authorisation Server implementing OIDC is also referred to as an **OpenID Provider (OPs).**
- A client which uses an OP for authentication is a **Relying Party (RPs)**.
- OIDC identifies a set of personal attributes that can be exchanged between Identity Providers and the apps that use them, and includes an approval step so that users can consent (or deny) the sharing of this information

Science and Technology Facilities Council

Want to know more about OIDC after the lecture

https://openid.net/connect/

OIDC: The ID Token

- In OAuth flows involving a user, the user will authenticate with the Authorisation Server before providing consent for the server to release information to the client. OpenID Connect utilises this process to authenticate to the client
- The Token Endpoint will provide two separate tokens: a standard OAuth Access Token, and the OIDC ID Token.
- The ID Token will contain information about the user pertaining to what the client has requested
- When the client receives the ID Token, it may then read off requested claims

OIDC: Requesting Info with Scopes

- When the client makes its token request, it must use **Scopes** to specify which privileges are being requested in the token
- In OAuth 2.0, Scopes correspond to what resources are available when a protected resource is accessed
- In OIDC, Scopes correspond to the specific sets of information to be made available as Claim Values
- OIDC defines a set of standard Scopes, but does allow additional Scope values to be defined and used

```
https://example.com/oauth/auth?
    response_type=code
    &scope=openid%20profile%20email
    &client_id=EXAMPLE_CLIENTID
    &redirect_uri=EXAMPLE_REDIRECT_URI
```

Science and Technology Facilities Council

OIDC: Default Scopes

as defined by OIDC Core

Claim	Summary
`openid`	REQUIRED. Informs the Authorization Server that the Client is making an OpenID Connect request. If the `openid` scope value is not present, the behaviour is entirely unspecified.
`profile`	OPTIONAL. This scope value requests access to the End-User's default profile Claims, which are: `name`, `family_name`, `given_name`, `middle_name`, `nickname`, `preferred_username`, `profile`, `picture`, `website`, `gender`, `birthdate`, `zoneinfo`, `locale`, and `updated_at`.
`email`	OPTIONAL. This scope value requests access to the `email` and `email_verified` Claims.
`address`	OPTIONAL. This scope value requests access to the `address` Claim
`phone`	OPTIONAL. This scope value requests access to the `phone_number` and `phone_number_verified` Claims.
`offline_access`	OPTIONAL. This scope value requests that an OAuth 2.0 Refresh Token be issued that can be used to obtain an Access Token that grants access to the End-User's UserInfo Endpoint even when the End-User is not present (not logged in).

OIDC: Default Token Required Claims

as defined by OIDC Core

```
{ "iss":
 "https://server.example.com",
 "sub": ”24400320",
 "aud": "s6BhdRkqt3”,
 "exp": 1311281970,
 "iat": 1311280970,
 "auth_time": 1311280969”
}
```

Claim	Summary
`iss`	REQUIRED: Issuer identifier for who issued the response. This is a case sensitive URL
`sub`	REQUIRED: Subject identifier. A locally unique identifier which must never be reassigned, this identifies the user within the issuer. A case sensitive string ≤ 255 ASCII characters in length
`aud`	REQUIRED: Audiences that this token is intended for, using the Client ID as its identifying value.
`exp`	REQUIRED: Expiration time for the token, after which it MUST NOT be used for processing.
`iat`	REQUIRED: Time at which the JWT was issued
`auth_time`	Time when end-user authentication occurred. If a `max_age` request is made or `auth_time` is requested as an essential claim, this is REQUIRED – otherwise it is OPTIONAL

Science and Technology Facilities Council

OIDC: Standard Identity Claims

as defined by OIDC Core

Claim	Summary
`name`	The end-user's full name, including all name parts. `given_name`, `family_name`, `middle_name` and `nickname` may also be used.
`preferred_username`	A shorthand name by which the user wishes to be referred to by at the RP. This MAY be any valid JSON string – including special characters such as @, /, or whitespace. The RP MUST NOT rely on this value being unique.
`email`	End-User's preferred e-mail address. Its value MUST conform to the RFC 5322 addr-spec syntax. The RP MUST NOT rely upon this value being unique.
`phone_number`	End-user's preferred telephone number.
`address`	End-User's preferred postal address.
`profile`	URL of the End-User's profile page. The contents of this Web page SHOULD be about the End-User.
… And others	`picture`, `website`, `email_verified`, `gender`, `birthdate`, `zoneinfo`, `locale`, `phone_number_verified`, `updated_at`

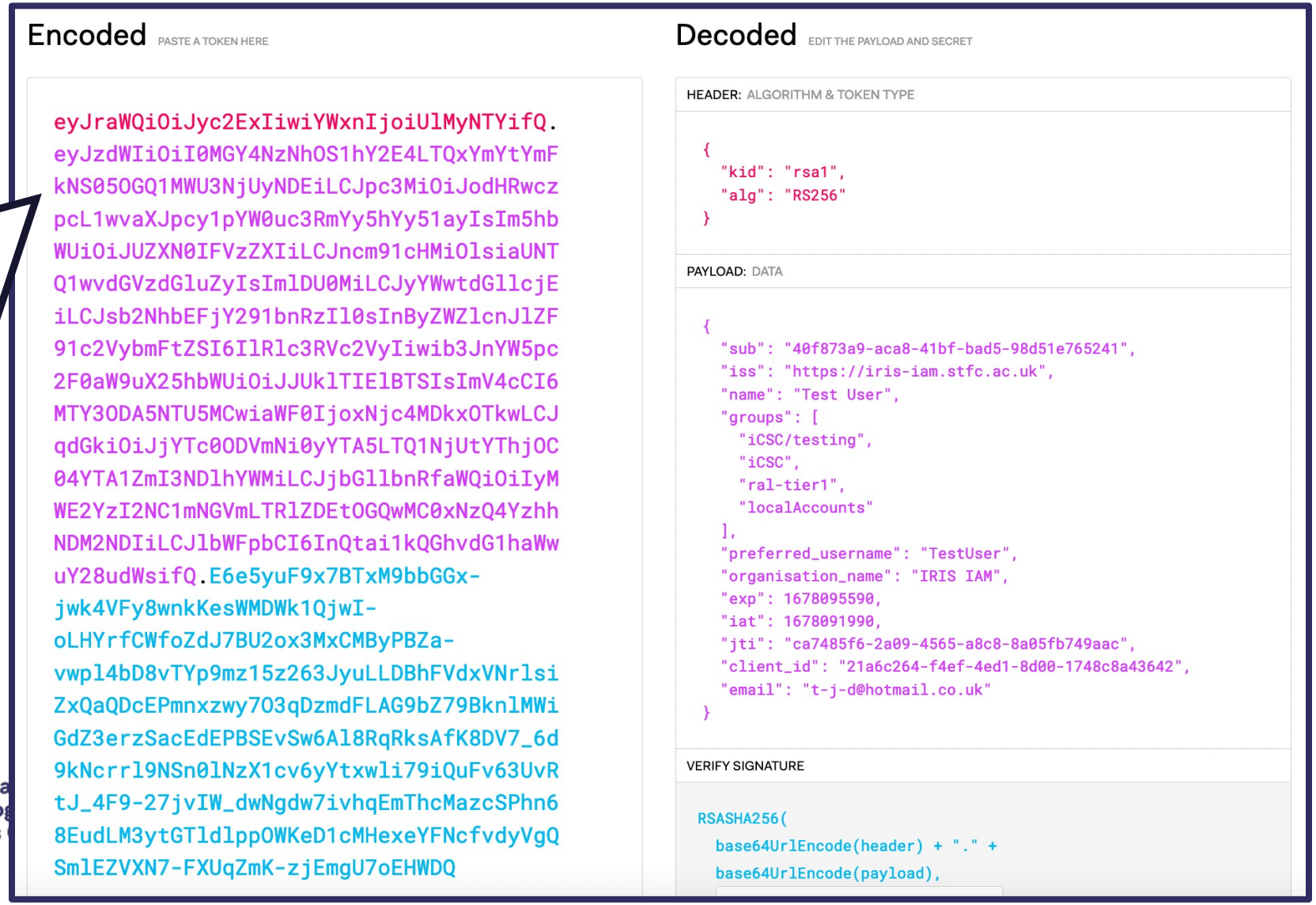
OIDC: Additional Claims

- The OpenID Connect Core specification only defines the previous small set of claims as standard
- However, OP's MAY provide additional claims about the End-User
- A typical example includes `groups`, a list of groups the user is registered with at the OP
- Any additional Claim will need to have a corresponding Scope, to allow it to be requested
  - Alternatively, extra claims could be included within the `profile` scope – this is how the OP to be used within the exercises provides its `groups` claim

Science and Technology Facilities Council

OIDC: Putting this together…

The token is three Base64-URL strings separated by dots that can be easily passed in HTML and HTTP environments



Handy token decoding taken from: https://jwt.io/

Check it out for more info on JWTs or when using them yourself!

OIDC: Putting this together…

eyJraWQiOiJyc2ExIiwiYWxnIjoiUlMyNTYifQ.

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

OIDC: Putting this together…


```
eyJzdWIiOiI0MGY4NzNhOS1hY2E4LTQxYmYtYmF
kNS05OGQ1MWU3NjUyNDEiLCJpc3MiOiJodHRwcz
pcL1wvaXJpcy1pYW0uc3RmYy5hYy51ayIsIm5hb
WUiOiJUZXN0IFVzZXIiLCJncm91cHMiOlsiaUNT
Q1wvdGVzdGluZyIsImlDU0MiLCJyYWwtdGllcjE
iLCJsb2NhbEFjY291bnRzIl0sInByZWZlcnJlZF
91c2VybmFtZSI6IlRlc3RVc2VyIiwib3JnYW5pc
2F0aW9uX25hbWUiOiJJUklTIElBTSIsImV4cCI6
MTY3ODA5NTU5MCwiaWF0IjoxNjc4MDkxOTkwLCJ
qdGkiOiJjYTc0ODVmNi0yYTA5LTQ1NjUtYThjOC
04YTA1ZmI3NDlhYWMiLCJjbGllbnRfaWQiOiIyM
WE2YzI2NC1mNGVmLTRlZDEtOGQwMC0xNzQ4Yzhh
NDM2NDIiLCJlbWFpbCI6InQtai1kQGhvdG1haWw
uY28udWsifQ.
```


Decoded EDIT THE PAYLOAD AND SECRET

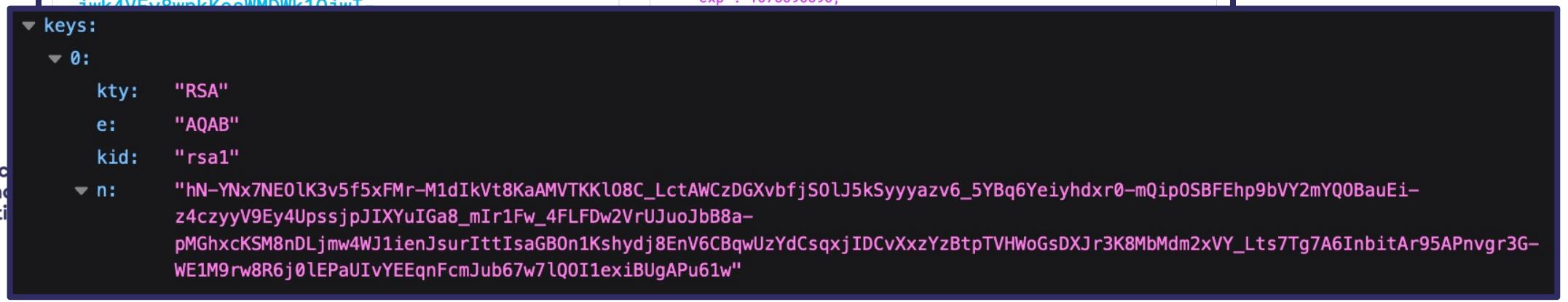

```
{
  "sub": "40f873a9-aca8-41bf-bad5-98d51e765241",
  "iss": "https://iris-iam.stfc.ac.uk",
  "name": "Test User",
  "groups": [
    "iCSC/testing",
    "iCSC",
    "ral-tier1",
    "localAccounts"
  ],
  "preferred_username": "TestUser",
  "organisation_name": "IRIS IAM",
  "exp": 1678095590,
  "iat": 1678091990,
  "jti": "ca7485f6-2a09-4565-a8c8-8a05fb749aac",
  "client_id": "21a6c264-f4ef-4ed1-8d00-1748c8a43642",
  "email": 🙂
}
```



```
9kNcrrl9NSn0lNzX1cv6yYtxwli79iQuFv63UvR
tJ_4F9-27jvIW_dwNgdw7ivhqEmThcMazcSPhn6
8EudLM3ytGTldlppOWKeD1cMHexeYFNcfvdyVgQ
SmlEZVXN7-FXUqZmK-zjEmgU7oEHWDQ
```


OIDC: Putting this together…

JWK signing key published at a URL at the OP, such as:
https://wlcg.cloud.cnaf.infn.it/jwk
The specific URL is detailed in the OP metadata, found at:
`.well-known/openid-configuration`
For Example:
https://wlcg.cloud.cnaf.infn.it/.well-known/openid-configuration



OIDC: UserInfo Endpoint

- The ID token is not the only way for a RP to get information about a user – it can also make a request to the Ops `UserInfo` endpoint
- A UserInfo Request is made by the client using either a HTTP `GET` or HTTP `POST`. The Access Token previously obtained from an OIDC Authentication request MUST be included as a Bearer Token
- The OIDC Core recommendation is that the request uses the HTTP `GET` method, with the Access Token sent using the `Authorization` header field

A non-normative example of a UserInfo Request:

```
GET /userinfo HTTP/1.1
Host: server.example.com
Authorization: Bearer SlAV32hkKG
```

Science and Technology Facilities Council

Science and Technology Facilities Council

Towards Tokens for WLCG

Transition to Tokens - Motivations

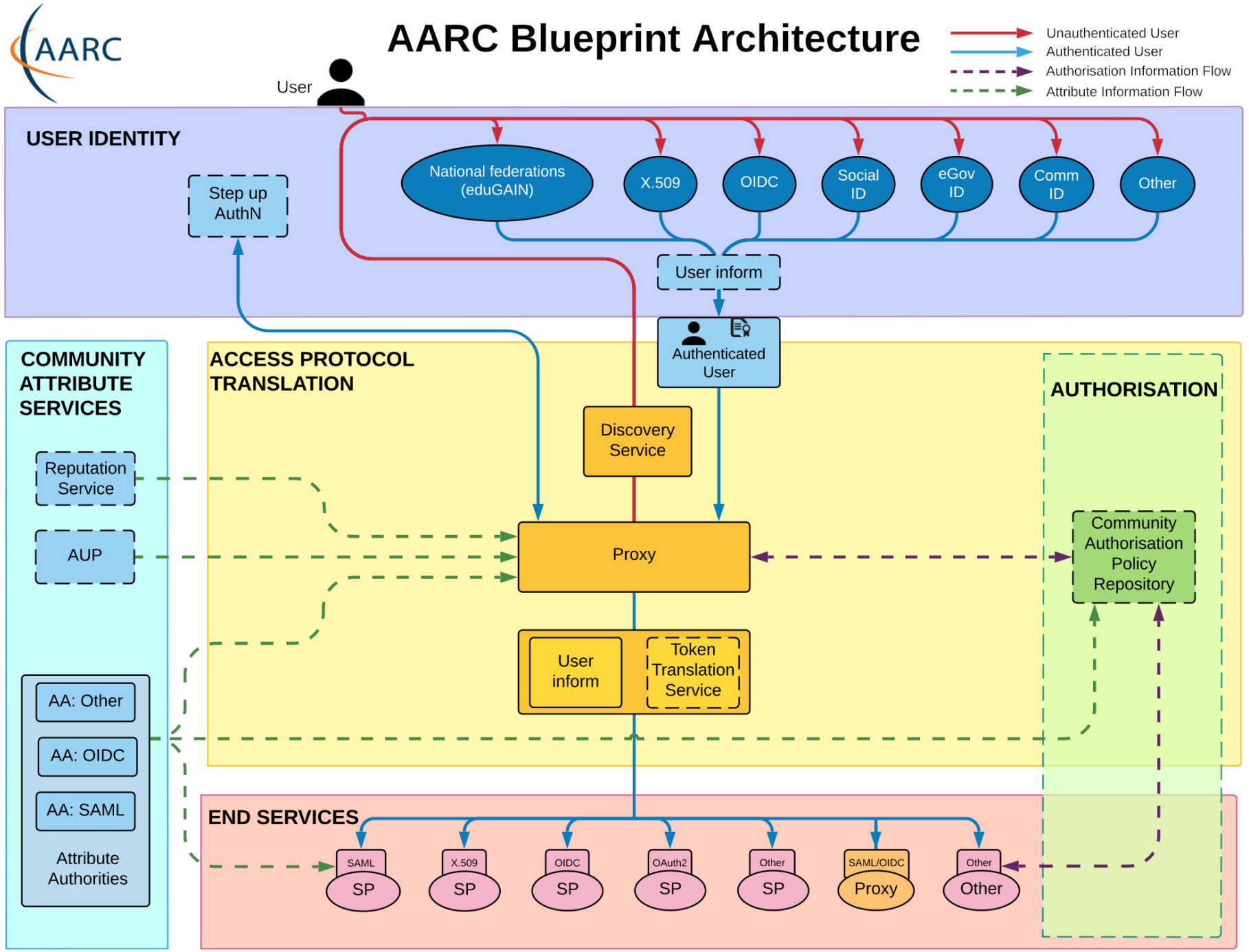
- OAuth and OIDC protocols have been adopted by a wide range of software and systems, thanks to their prevalence in industry – in particular within the social identity space
  - This prevalence enables developers to utilise developed libraries, and facilitates integration and interoperability

- Certificates, whilst established within WLCG, are not familiar outside this context - and are often viewed as unintuitive. Conversely, Token-based flows are becoming increasingly commonplace, and can lead to a better user experience as a result

Token Authentication and Authorisation Infrastructure

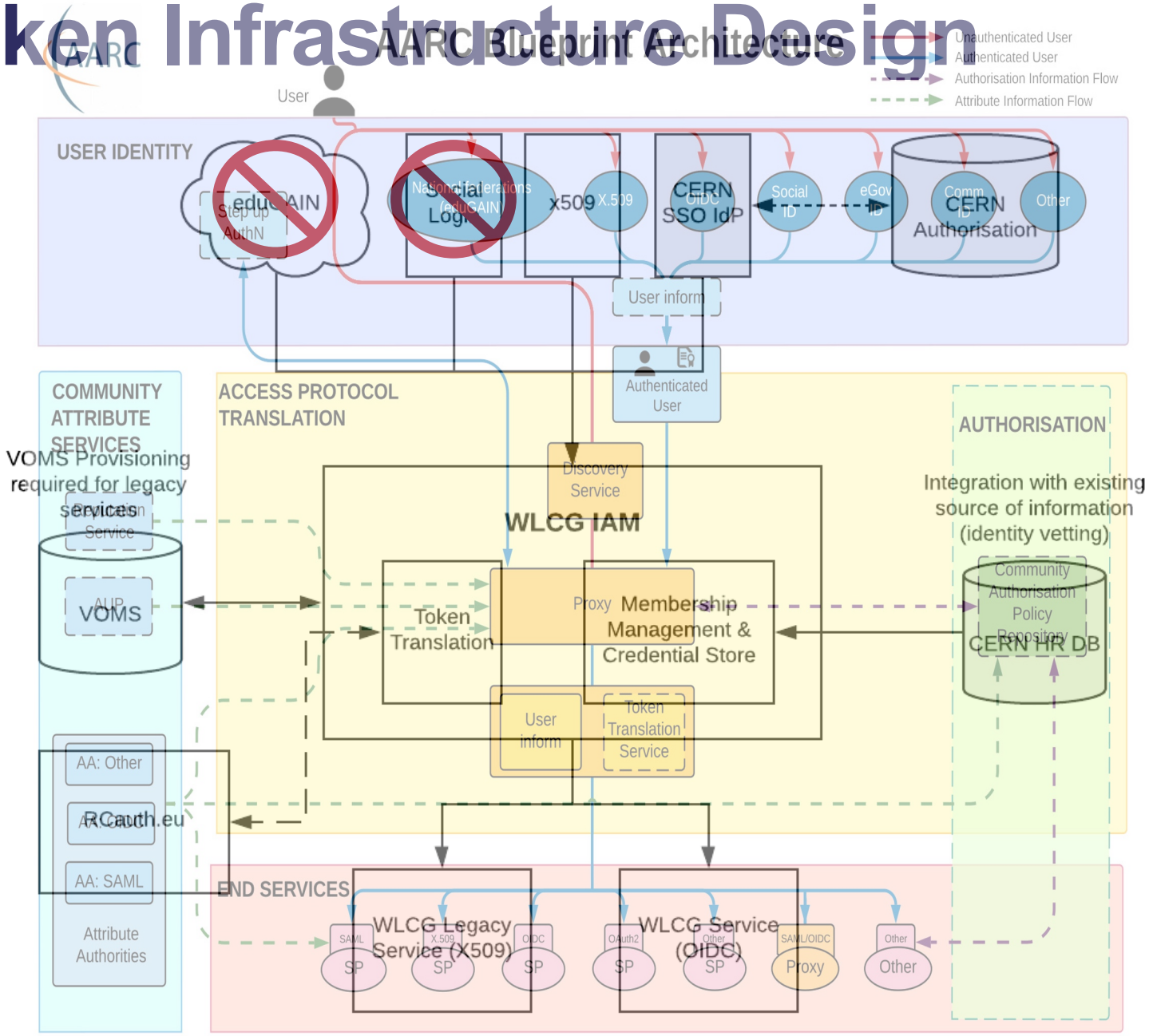
- In the planned WLCG infrastructure, there will be an OP per VO, which users may access using their CERN Account
- This model unifies the IdP and Research community, with both being represented by the Token Issuer
- The Infrastructure design has been informed by the **AARC Blueprint architecture** - a set of software building blocks that can be used to implement federated access management solutions for international research collaborations

The AARC Blueprint

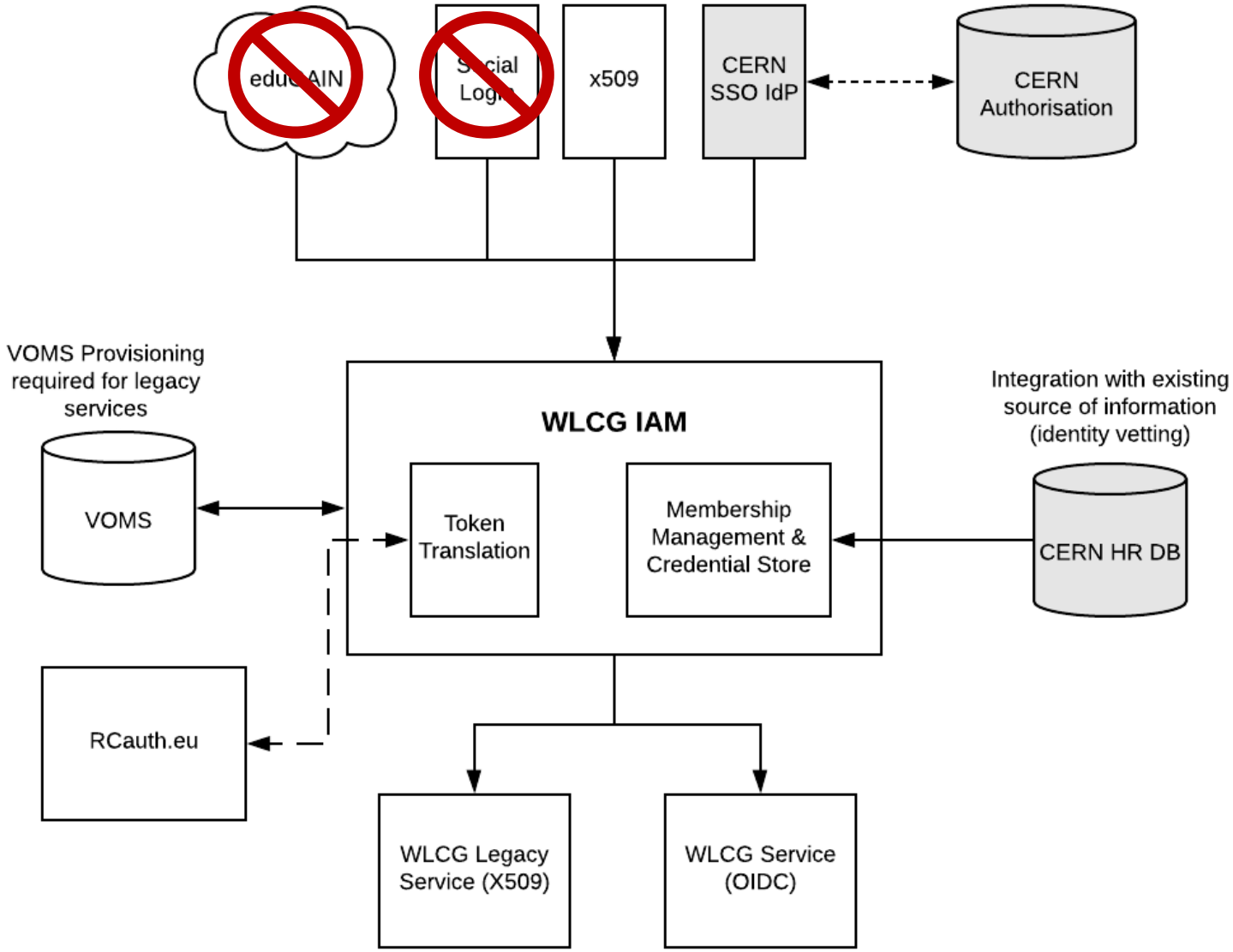


Science and Technology Facilities Council

WLCG Token Infrastructure Design



WLCG Token Infrastructure Design



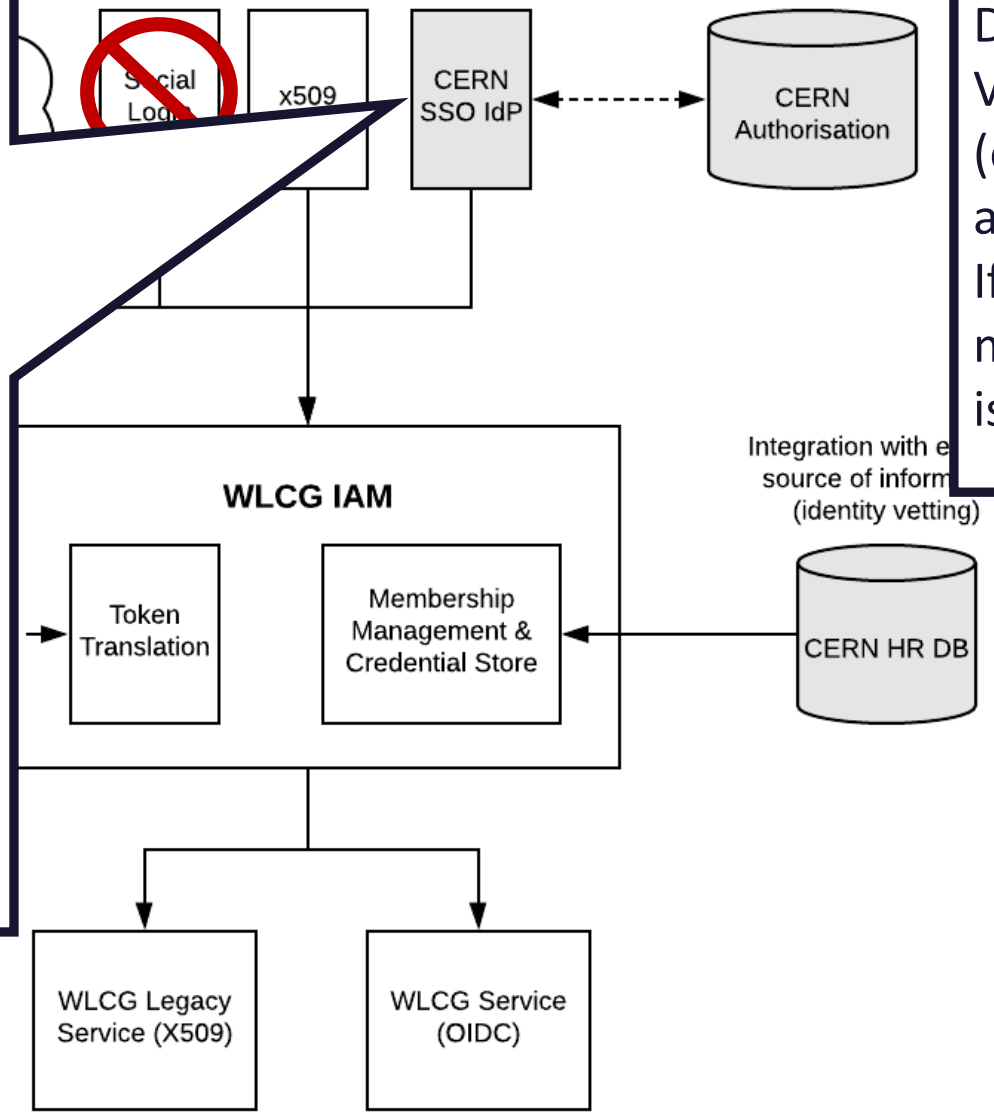
Science a
Technolog
Facilities C

WLCG Token Infrastructure Design

CERN SSO releases:

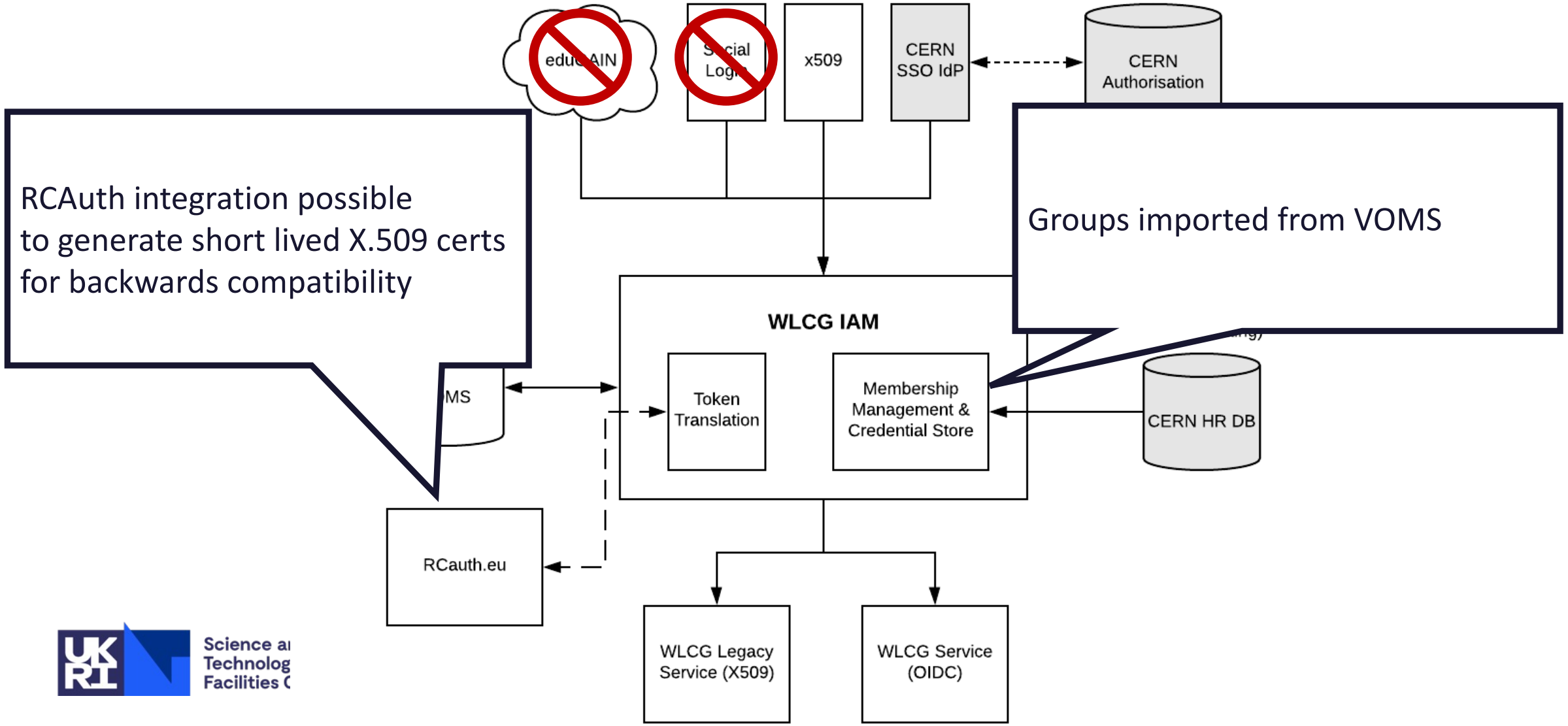
- Name,
- Email,
- CERN Person ID (indicates HR has performed ID check),
- CERN Kerberos Principal
- ...

Currently all researchers have CERN accounts but aim is to work towards removing this need in future



CERN Person ID is checked against CERN HR DB. Affiliation with Virtual Organisation (experiment) is verified, as well as end dates.
If the check is OK, the membership is approved.

WLCG Token Infrastructure Design



Science a
Technolog
Facilities C

WLCG Token Schema

- In order to serve authorisation information a VO, the WLCG token schema definies extra claims – `wlcg.groups` for Group-based authorisation and `scopes` for Capability-based authorisation
- `wlcg.groups` semantics are equivalent to existing VOMS groups, and will be initially imported directly from VOMS
  - Eg: /atlas/production
- `scopes` is used to provide capability to a specific token, rather than permanent authorisation to a user
  - Format `$AUTHZ:$PATH` where `$PATH` is mandatory (may be '/' for *)
  - Eg: storage.read:/atlas
- For more details, you can see the published schema: https://zenodo.org/record/3460258#.Y-YqUxPMLVs

Science and Technology Facilities Council

Science and
Technology
Facilities Council

Questions?

UKRI Science and Technology Facilities Council

Thank you

Science and Technology Facilities Council @STFC_matters Science and Technology Facilities Council