SRCNET USER MANAGEMENT SERVICE (UMS)

One Identity, Many Local Nodes: Identity Federation for SRCNet with OIDC

Adrian Damian, IVOA Interop, Perth, Nov 7 2026

canSRC / SKA · CANFAR / NRC

OVERVIEW

Outline

1. **Interop 2025** Continuation of a presentation at Interop 2025.
2. **Federation Goals** Single sign-on, global vs local identity.
3. **Implementation** Keycloak extension, Sequence flows

4 OIDC terms and actors used throughout

IdP OpenID Provider; authenticates the user (SKA IAM)

client Receives the SKA IAM login and calls a protected resource

iss Issuer — URL of the server that minted the token

sub Subject — stable identifier of the user at that issuer

aud Audience — resource URL the token is bound to

JWT Signed access token a service validates locally

token exchange New token, same subject, different audience

PKCE Ties the authorization-code flow to the public client

Protected Resource Storage or compute service at the site

GMS Group service: is this subject in the group?

posix-client Service account; client credentials for /uid and /gid

THE PROBLEM

Multiple identities across a federated network

Globally distributed centres, joined in a federation

These are networks of data and computing centres, designed to store, process, and analyze massive petabyte-scale datasets. SKA SRCNet is one example.

Each centre must support local users as well as users from other external networks in the federation. One person therefore has more than one identity.

1 Local users

People who belong to this centre — staff, admin-created, or self-registered. They receive a POSIX identity and use the site's services.

2 Users from other networks

People whose identity was issued by an external network, such as SKA IAM. Each centre chooses which federations it joins.

3 Same person, many centres

Alex@SKA is ska-alex at Site A and ska-alex at Site B. Same person, a separate local account and uid at each centre.

4 One local account, several networks

At Site A, one account can link to both Alex@SKA and Alex@Rubin. Prefixes (ska-, rubin-) keep those names distinct from local users.

CONCEPT

Global identity vs local identity

Global identity — SKA IAM

- One identity per person across SRCNet
  - INDIGO IAM: ska-iam.stfc.ac.uk
- Authenticates users (login, federation, MFA)
- Issues tokens: iss, sub, preferred_username, data rights
- Audience-scoped tokens per service
- No knowledge of site UIDs, GIDs, home directories, or even local services

brokered link



iss + sub

Local identity — UMS (Keycloak at the site)

- Site account linked to the global identity
  - created automatically on first SKA IAM login
- POSIX identity: username, uidNumber, gidNumber, home, shell
- Local token issuer: tokens carry posix.* claims
- Global identity reference
- Works with a local GMS for group-based authorization
- Identity known to local services

SKA IAM answers **“who are you?”** UMS answers **“who are you here?”** and issues the tokens site services trust.

What we customised in Keycloak

Provider JAR · ums-keycloak-posix

Event listener	skao-posix — POSIX account on first login / create
Realm REST	/uid /gid /capabilities (VOSI discovery)
LDAP mapper	skao-posix-account → writable posixAccount
User profile	posix.* attributes visible and editable
Validators	format, uniqueness, reserved prefixes (ska-*)

Declarative realm & server config

Identity provider SKAIAM
OIDC broker; preferred_username → posix.username

Client scope posix
Protocol mappers: posix.* → access-token claims

posix-client
Service account (client credentials) for /uid and /gid

oidc-exchange-client
JWT-bearer grant (IdP = SKAIAM), token exchange, device

oidc-client
Public PKCE client for CLI / browser login

No fork. Stock Keycloak 26.7.3 + provider JAR + kc.sh build. Config in keycloak.conf and realm import, shipped as Helm chart charts/ums.

SSO initiated at SKA IAM — and the specs involved

1	User → SKA IAM	Authenticate	OIDC Core · Discovery
2	SKA IAM → client	initiate_login_uri	OIDC Core §4
3	Browser → client client → SKA IAM	Authorization code, redeemed with PKCE	RFC 6749 · RFC 7636
4	SKA IAM → client	JWT, iss = SKA IAM	OIDC Core · RFC 7519
5	client → Protected Resource	Call with the SKA IAM token	RFC 6750 Bearer
6	Protected Resource → client	401 ivoa_bearer + resource_metadata	AuthVO · RFC 9728 §5
7	client → Protected Resource	GET metadata → authorization_servers: [UMS]	RFC 9728
8	client → UMS	Check resource ∈ protected_resources	RFC 8414 · RFC 9728 §4
9	client → UMS	jwt-bearer exchange → local JWT, aud = resource URL	RFC 7523 · RFC 8707

Result: one login at SKA IAM. The client holds a UMS-issued, audience-bound JWT. Logout across IdP and UMS is still open.

Service side: exchange the aud-scoped token for GMS

10	client → Protected Resource	Bearer JWT, aud = https://storage.site/	RFC 6750 · RFC 8707
11	Protected Resource, *locally*	Check iss, aud, signature, exp	RFC 7519 · no introspection
12	Protected Resource → UMS	Token exchange, audience = GMS URL	RFC 8693 exchange
13	UMS → Protected Resource	New JWT, same sub, aud = https://gms.site/	RFC 7519 · RFC 8707
14	Protected Resource → GMS	Is this subject in the required group?	IVOA GMS
15	GMS → Protected Resource Protected Resource → client	Membership, then 200 or 403	group-based authz

Why exchange: an aud-bound token for the resource must not be accepted by GMS. Same pattern as any other service-to-service call. Applies to federated and local-only users. /gid maps the group name to a POSIX GID; membership itself is GMS.

Assumptions this design rests on

JWT access tokens

Self-contained (RFC 7519). Protected resources validate signature, iss, exp, aud against the issuer JWKS. No RFC 7662 introspection — that would couple every call to the AS.

Audience-bound to a network location

aud is the resource URL / origin (RFC 8707). A token for https://storage.site/ is rejected by https://compute.site/. Prevents replay across services.

Group-based authorization + GMS

UMS issues identity and POSIX mapping. An IVOA GMS holds group membership. UMS /gid maps a GMS group URI onto a numeric GID. Decisions are “is this subject in this group?”

Token renewal

Little consideration given to renewal. Bearer tokens last minutes or hours. Jobs can last days.

UMS production deployment

Keycloak must be deployed on a high-availability backend (LDAP or a database). uid/gid mappings are part of authentication.

Also assumed

TLS everywhere; loosely synced clocks; OIDC discovery at SKA IAM and at UMS; clients honour ivoa_bearer + RFC 9728; issuer-prefixed usernames; one primary GMS per site today; UMS is an OIDC client of each federation IdP.

Closing remarks

A prototype

This is a prototype. Still work in progress — there is no GitHub or GitLab link to share yet.

A hard technical problem

Prototype, test, re-work. It is the only way to tackle it.

Also a policy challenge

How can **TRUST** between the nodes and global be established and maintained? User identity, tracebility, solid procedure in place (auth revocation, audit, etc.).

Thank you

SRCNet UMS · Keycloak as a local identity & token issuer

