Large-scale EXecution for Industry & Society

Deliverable D4.7

Centralized AAI: Coverage of All Significant Systems

Co-funded by the Horizon 2020 Framework Programme of the European Union
Grant Agreement Number 825532
ICT-11-2018-2019 (IA - Innovation Action)

DELIVERABLE ID | TITLE	D4.7 | Centralized AAI: Coverage of All Significant Systems
RESPONSIBLE AUTHOR	Frédéric Donnat (O24)
WORKPACKAGE ID | TITLE	WP4 | Orchestration and Secure Cloud/HPC Services Provisioning
WORKPACKAGE LEADER	LINKS
DATE OF DELIVERY (CONTRACTUAL)	30/09/2021 (M33)
DATE OF DELIVERY (SUBMITTED)	17/10/2021 (M34)
VERSION | STATUS	V1.0 | Final
TYPE OF DELIVERABLE	R (Report)
DISSEMINATION LEVEL	PU (Public)
AUTHORS (PARTNER)	O24, Atos, LRZ, IT4I
INTERNAL REVIEW	Marc Levrier (Atos), Lukáš Vojáček (IT4I)

Project Coordinator: Dr. Jan Martinovič – IT4Innovations, VSB – Technical University of Ostrava
E-mail: jan.martinovic@vsb.cz, **Phone:** +420 597 329 598, **Web:** https://lexis-project.eu

DOCUMENT VERSION

VERSION	MODIFICATION(S)	DATE	AUTHOR(S)
0.1	Table of Contents and Partners´ involvement	02/06/2021	Frédéric Donnat (O24)
0.1	Update table of Content with partners´ feedback	07/06/2021	Frédéric Donnat (O24)
0.1	First version of DDI part from LRZ	29/06/2021	Stephan Hachinger (LRZ)
0.1	Revised version of DDI part from LRZ	06/07/2021	Rubén García Hernández (LRZ)
0.2	Adding short introduction and description, links with others deliverables and introducing all parts	13/09/2021	Frédéric Donnat (O24)
0.3	Adding missing content on Keycloak and several parts for linking the whole document.	22/09/2021	Frédéric Donnat (O24)
0.4	Global proof reading, review	27/09/2021	Marc Levrier (Atos), Lukáš Vojáček (IT4I)
0.5	Handling reviewer's comments and removing 'TODO' tasks	30/09/2021	Frédéric Donnat (O24)
0.6	Correcting references of Section 3	11/10/2021	Frédéric Donnat (O24), Rubén García Hernández (LRZ)
1.0	Final check of the deliverable	14/10/2021	Kateřina Slaninová (IT4I)

GLOSSARY

ACRONYM	DESCRIPTION
AAI	Authentication and Authorization Infrastructure
ABAC	Attribute-Based Access Control
ACL	Access Control List
API	Application Programming Interface
DDI	Distributed Data Infrastructure
IAM	Identity and Access Management
HA	High Availability
HPC	Hyper-Performance Computing
IAM	Identity and Access Management
PBS	Portable Batch System
RBAC	Role-Based Access Control

REST	Representational State Transfer
SLURM	Simple Linux Utility for Resource Management
SSH	Secure SHell
SSL	Secure Socket Layer
YORC	Ystia ORChestrator

TABLE OF PARTNERS

ACRONYM	PARTNER
Avio Aero	GE AVIO SRL
Atos	BULL SAS
AWI	ALFRED WEGENER INSTITUT HELMHOLTZ ZENTRUM FUR POLAR UND MEERESFORSCHUNG
BLABS	BAYNCORE LABS LIMITED
CEA	COMMISSARIAT A L ENERGIE ATOMIQUE ET AUX ENERGIES ALTERNATIVES
CIMA	CENTRO INTERNAZIONALE IN MONITORAGGIO AMBIENTALE - FONDAZIONE CIMA
CYC	CYCLOPS LABS GMBH
ECMWF	EUROPEAN CENTRE FOR MEDIUM-RANGE WEATHER FORECASTS
EURAXENT	MARC DERQUENNES
GFZ	HELMHOLTZ ZENTRUM POTSDAM DEUTSCHESGEOFORSCHUNGSZENTRUM GFZ
ICHEC	NATIONAL UNIVERSITY OF IRELAND GALWAY / Irish Centre for High-End Computing
IT4I	VYSOKA SKOLA BANSKA - TECHNICKA UNIVERZITA OSTRAVA / IT4Innovations National Supercomputing Centre
ITHACA	ASSOCIAZIONE ITHACA
LINKS	FONDAZIONE LINKS / ISTITUTO SUPERIORE MARIO BOELLA ISMB
LRZ	BAYERISCHE AKADEMIE DER WISSENSCHAFTEN / Leibniz Rechenzentrum der BAdW
NUM	NUMTECH
O24	OUTPOST 24 FRANCE
TESEO	TESEO SPA TECNOLOGIE E SISTEMI ELETTRONICI ED OTTICI

TABLE OF CONTENTS

EXECUTIVE SUMMARY **6**

1 INTRODUCTION OF LEXIS AAI AND HPC IDENTITIES **7**

1.1 GOAL 7
1.2 CONSTRAINTS 7
1.2.1 Heterogeneous local security policies *7*
1.2.2 No direct connection to HPC authentication backend services *7*
1.3 RELATION TO IAM OF HPC 7

2 LEXIS AAI **9**

2.1 DEPLOYMENT: CROSS DATACENTER REPLICATION MODE 9
2.1.1 Galera Database Cluster based on MariaDB *10*
2.1.2 Cache cluster based on Infinispan *10*
2.1.3 IAM Cluster based on Keycloak *10*
2.2 CONFIGURATION 11
2.2.1 Keycloak Master & LEXIS AAI Realms *12*
2.2.2 Keycloak Client for each LEXIS Component *14*
2.2.3 RBAC implementation using ABAC *16*
2.2.4 Keycloak Library extension *17*

3 LEXIS DDI AND EMBEDDED IAM **18**

3.1 REDUNDANT DEPLOYMENT, DDI ZONING AND DATA MIRRORING 18
3.1.1 iRODS IAM *19*
3.2 DDI CONFIGURATION WITH RESPECT TO AUTHENTICATION & AUTHORISATION 20
3.2.1 Authenticating LEXIS DDI/iRODS users via OpenID Connect *20*
3.2.2 LEXIS AAI to LEXIS DDI identity and access mapping, RBAC implementation *20*
3.2.3 OpenID Connect usage and RBAC implementation within the LEXIS DDI APIs *22*

4 LEXIS MIDDLEWARE AND HPC, CLOUD IDENTITIES **22**

4.1 HEAPPE MIDDLEWARE 22
4.1.1 HEAppE Identities *23*
4.1.2 HPC and Cloud Identities *23*
4.1.3 LEXIS AAI to HPC and Cloud identities and access mapping *23*
4.1.4 OpenIDC for LEXIS AAI *24*
4.2 CONFIGURATION 24

5 CONCLUSION **25**

REFERENCES **26**

LIST OF FIGURES

FIGURE 1 MANAGING IDENTITIES AND ACCESSES WITHIN LEXIS PLATFORM....8
FIGURE 2 HIGH-LEVEL SCHEMA OF LEXIS AAI ARCHITECTURE....9
FIGURE 3 LIFECYCLE OF THE ACCESS TOKEN....11
FIGURE 4 LOGIN TO KEYCLOAK....12
FIGURE 5 USER LOGIN - LEXIS REALM....12
FIGURE 6 IDENTITY PROVIDERS CONFIGURED IN KEYCLOAK MASTER REALM....13
FIGURE 7 CONFIDENTIAL KEYCLOAK CLIENT DEPLOYED IN LEXIS AAI REALM....13
FIGURE 8 KEYCLOAK ROLE MAPPINGS....13
FIGURE 9 LEXIS AAI REALM ROLES....14
FIGURE 10 KEYCLOAK CLIENTS FOR LEXIS COMPONENTS....14
FIGURE 11 KEYCLOAK CLIENT MAPPERS TO PROPAGATE USER ATTRIBUTES TO LEXIS COMPONENTS....15
FIGURE 12 MAPPER CONFIGURATION FOR AN ORGANIZATION LISTING MAPPERS....15
FIGURE 13 KEYCLOAK HIGH-LEVEL GROUPS....16
FIGURE 14 KEYCLOAK: SUB-GROUPS IN ORGANIZATION GROUP....16
FIGURE 15 KEYCLOAK: KEY:VALUE ATTRIBUTE IN ORGANIZATION SUB-GROUP....17
FIGURE 16 KEYCLOAK: SUB-GROUPS IN PROJECTS GROUP....17
FIGURE 17 KEYCLOAK: KEY:VALUE ATTRIBUTE IN PROJECTS SUB-GROUP....17
FIGURE 18 LEXIS DDI FEDERATION CONCEPT....18
FIGURE 19 REDUNDANT IRODS-SERVER SETUP FOR ONE SITE IN THE LEXIS DDI. BOXES REPRESENT VIRTUAL MACHINES....19
FIGURE 20. IRODS-OPENID-CONNECT AUTHENTICATION FLOW WITH TOKEN PRE-VALIDATION ON CLIENT SIDE, HASHED TOKEN FLOW, AND HASH-JWT LOOKUP/VALIDATION IN AUTH_MICROSERVICE. 20
FIGURE 21 HPC AND CLOUD IDENTITY MAPPING....24

EXECUTIVE SUMMARY

This document describes a centralized Authentication and Authorization Infrastructure that is currently deployed on the LEXIS platform in order to provide it with Identity and Access Management (IAM).

As a matter of fact, the LEXIS Centralized Authentication and Authorization Infrastructure (AAI) is based on:

- LEXIS AAI distributed installation of IAM system, that is deployed in main HPC Supercomputing centres (IT4I and LRZ) providing source of truth for all identity and access of LEXIS Platform,
- LEXIS DDI deployed in each main HPC Supercomputing centre (IT4I and LRZ) and any other centre part of LEXIS DDI federation providing access to a LEXIS data layer,
- LEXIS Middleware based on HEAppE security middleware deployed on each main HPC Supercomputing centre (IT4I and LRZ) and any other centre providing computing capacity (HPC or Cloud).

This document contains a detailed description of the deployment and configuration of the above systems and the synchronization between all of them using LEXIS AAI as source of truth for both identity and access.

Position of the deliverable in the whole project context

This deliverable is following the preliminary study (containing design and implementation) of "Identity and Access Management" solution described in Deliverable D4.1 "Analysis of mechanisms for securing federated infrastructure" [1] as this is one pillar of security. This document also provides concrete description on how we implemented the concept of "Zero-Trust" architecture with LEXIS Platform and the security mechanisms that are described in Deliverable D4.5 "Definition of Mechanisms for Securing Federated Infrastructures" [2].

Description of the deliverable

The document starts with introducing a short recap of the goals and constraints related to IAM system within the context of a federated supercomputing platform.

Then the document describes deployment and configuration of all components involved in the authentication and authorization mechanisms for the LEXIS Platform, starting from LEXIS AAI (source of truth) and encompassing both LEXIS Distributed Data Infrastructure (DDI) and LEXIS Middleware.

Contributors to the deliverable content are:

- O24 as a leader of activities regarding security aspects, responsible for this deliverable,
- IT4I both as a HPC centre in charge of deploying and administrating LEXIS infrastructure (connecting to HPC systems), and as a provider of HEAppE software in charge of mapping LEXIS infrastructure security policies to the existing HPC (and Cloud-Computing) infrastructure,
- LRZ both as a HPC centre (analogous to IT4I), and as a WP3 leader in charge of the LEXIS Distributed data infrastructure (DDI),
- Atos both as a WP2 leader in charge of co-design activities, and a technology provider (for YORC and Alien4Cloud software that are used for the LEXIS orchestration).

1 INTRODUCTION OF LEXIS AAI AND HPC IDENTITIES

1.1 GOAL

The goal of this deliverable is twofold:

- Describe "LEXIS AAI" that is the centralized AAI system in LEXIS platform used for authentication and authorization of any user of the LEXIS Platform,
- Describe all other LEXIS components such as LEXIS DDI and LEXIS Middleware, that require a local/specific IAM system to authenticate and authorize the LEXIS users.

1.2 CONSTRAINTS

The LEXIS project being a federation of HPC supercomputing centre and cloud environment, it implies being able to handle different security policies, federating different existing IAM solutions with different constraints.

1.2.1 Heterogeneous local security policies

In the context of the LEXIS project, federating several data centres for identity and access management, also means federating several types of access to different IAM systems that do not rely on the exact same security constraints.

For example, one supercomputing centre may give a read-only access to their internal identity management system whereas the other supercomputing centre may not give any access to any third-party application not fully running in their premises. This difference is usually due to the different security policies or constraints and are motivated by preventing identity leakage from any software. As a matter of fact, if any third-party software installed in a different data centre is retrieving identities from your directory server, then it is hard to control that any identity will not leak from that third-party software as you do not control it.

1.2.2 No direct connection to HPC authentication backend services

Moreover, supercomputing centres usually have directory servers without any SAML or OpenID interfaces, thus involving creating some service account to give access to third-party software.

The main constraint for the LEXIS AAI is that it cannot rely on being able to access supercomputing centre directory server to access identities, but it needs to contain specific LEXIS identities that will be mapped later on to supercomputing centre identities thanks to the HEAppE Middleware.

1.3 RELATION TO IAM OF HPC

Usually, supercomputing centres rely on traditional directory servers with Kerberos interface to provide identity management. Most of the time, they have deployed some Microsoft Active Directory or LDAP clusters in order to manage their users and groups. This is also related to the fact that they usually need easy integration with Unix-Like or Linux-Like systems that are the core of any HPC infrastructure. This way they can easily link the supercomputing infrastructure with the directory system by using some Linux Pluggable Authentication Module (PAM) or some Public Key Infrastructure based access (such as SSH or Certificate).

Those traditional directory servers are rarely extended with a federation deployment that would provide SAML or OpenIDC connectivity to any third-party software. Usually, a service account is created in order to give access to such third-party software which can be a security concern as it gives the third-party software some access privilege that can then be exploited to at least list all identity.

As a side note, it can be even worse when there is no secure communication channel between the third-party software and the directory server, as it exposes the service account password to leak over the network.

In the context of the LEXIS Project, we need to manage and synchronize identities and accesses between several places for several components:

- LEXIS Platform identities and accesses are managed by LEXIS AAI and underlying Keycloak IAM tool,
- LEXIS DDI identities and accesses are managed by iRODS through a set of system users, permissions and maps via OpenIDC integration for iRODS,
- LEXIS HPC and Cloud identities are managed locally in each supercomputing centre and mapped using HEAppE middleware and OpenIDC integration.

Those different systems managing identities and accesses within LEXIS Platform are described on the schema in Figure 1.

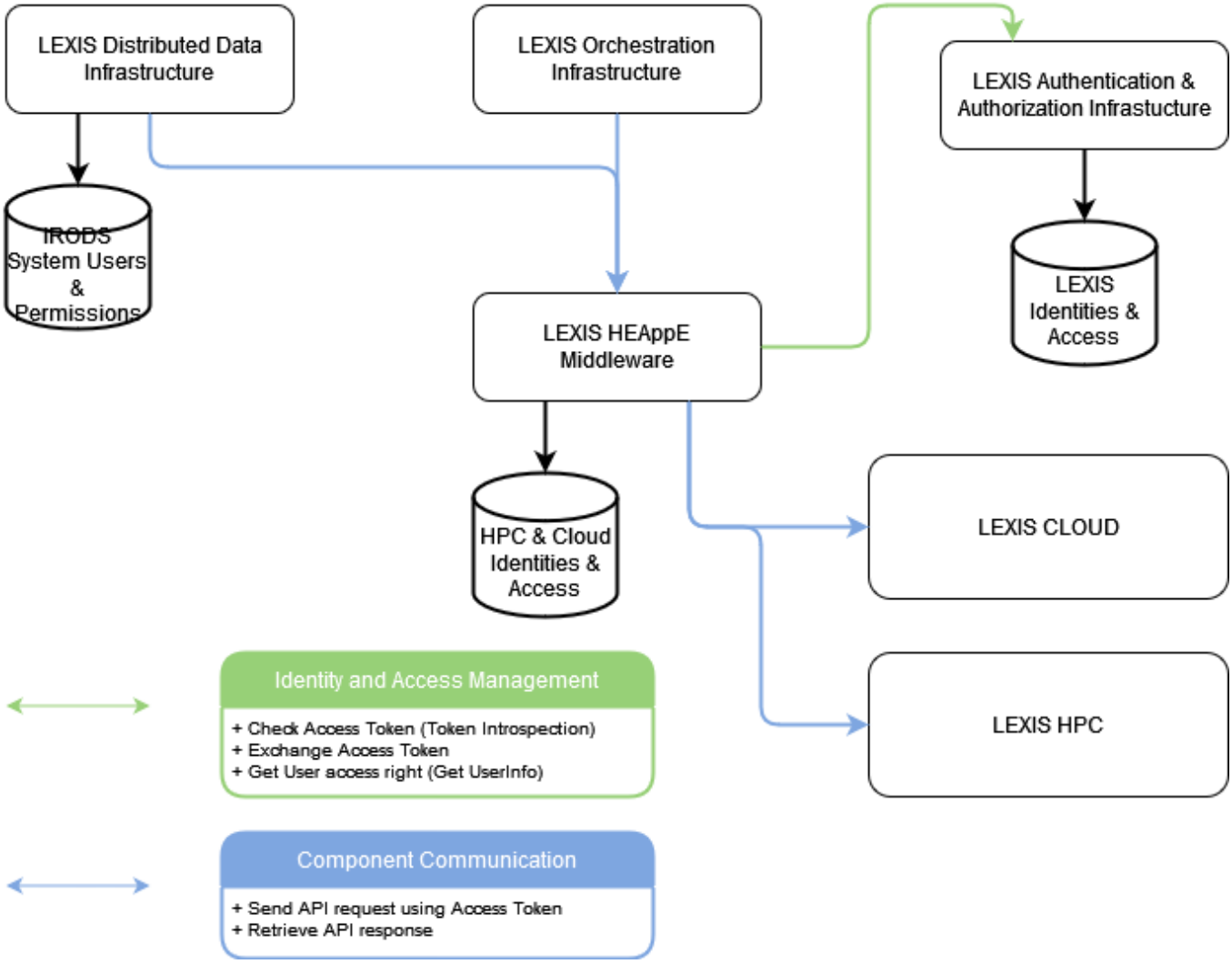

Figure 1 Managing identities and accesses within LEXIS platform

2 LEXIS AAI

LEXIS AAI encompasses all the software that is deployed in supercomputing centre to provide authentication and authorization for the LEXIS Platform.

It is worth to mention that all the software is updated on regular basis or when a security issue impacting our deployment and usage is disclosed.

2.1 DEPLOYMENT: CROSS DATACENTER REPLICATION MODE

Please refer to the high-level schema in Figure 2 to understand the LEXIS AAI architecture that has been described in details in Deliverable D4.1 [1] and Deliverable D4.5 [2].

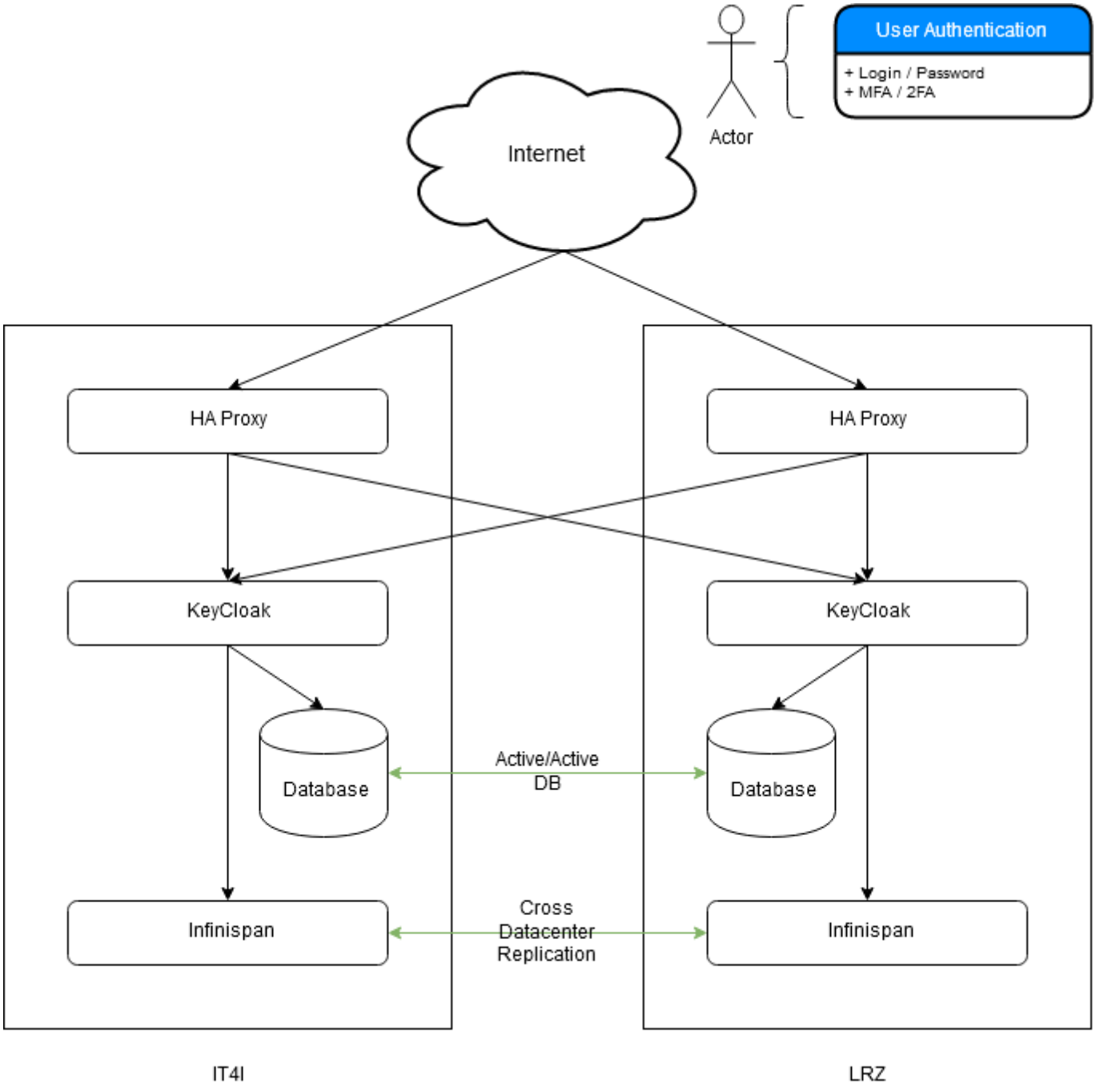

Figure 2 High-level schema of LEXIS AAI architecture

It is worth to mention that the LEXIS AAI and all its software are deployed on virtual machines running on IT4 and LRZ data centre on 2 different LINUX operating systems. IT4I is using CentOS Linux 7 as base operating system whereas LRZ is using Ubuntu 18.04.6 LTS as base operating system.

A local host-based firewall (based on Linux iptables) is configured on all virtual machines to only allow connections accordingly to the above architecture diagram.

HAProxies are configured in cluster mode in each supercomputing centre and the global LEXIS DNS servers have been configured to redirect equally to both supercomputing centres.

As LEXIS AAI relies on Keycloak open-source IAM solution, the infrastructure must provide the following data sources for the stateful application.

2.1.1 Galera Database Cluster based on MariaDB

A database cluster is required to store persistent data such as identity information (mainly users), grouping information (such as user's group) and access granting.

The database cluster is deployed in an active/active mode in both IT4I and LRZ data centres. This database cluster currently relies on MariaDB Galera Cluster version 10.3.29 that encompasses Galera write set replication (wsrep) version 25.3.33 for Galera 3.

The database cluster is configured to use the site-2-site VPN between IT4I and LRZ for data replication and synchronization between the nodes of the cluster.

The cluster is configured in an active/active mode with 1 node in each supercomputing centre deployed on a quite small virtual machine with 2 CPUs, 4GB RAM and 60 GB disk shared with cache cluster.

The cluster can be scaled up by adding node in each supercomputing centre according to the needs.

2.1.2 Cache cluster based on Infinispan

The cluster cache is used for both cache persistent data and storage of ephemeral/short-lived data (such as user's session) or frequently changing meta-data. As a side note, the cache is non persistent upon restart of the cluster.

Two cache clusters are deployed in an active/active mode in both IT4I and LRZ supercomputing centre. Those clusters currently rely on Infinispan version 9.4.20.

The cache clusters are configured to use the site-2-site VPN between IT4I and LRZ for data replication and synchronization between the cluster.

The clusters are configured in an active/active mode with 1 node in each supercomputing centre deployed on a quite small virtual machine with 2 CPUs, 4GB RAM and 60 GB disk shared with the database cluster node.

The clusters can be scaled up by adding node in each supercomputing centre according to the needs.

2.1.3 IAM Cluster based on Keycloak

Two Keycloak clusters have been deployed in an active/active mode in both IT4I and LRZ supercomputing centres. Those clusters are currently relying on Keycloak version 12.0.4.

The Keycloak clusters are configured to use the database and cache from their respective supercomputing centre.

The clusters are configured in high-availability mode with 1 node in each supercomputing centre deployed on a quite small virtual machine with 2 CPUs, 4GB RAM and 60 GB disk.

The clusters can be scaled up by adding nodes in each supercomputing centre according to the needs.

2.2 CONFIGURATION

Each LEXIS service/component has a “Keycloak Client” configured, so that each component can:

- Validate the user provided access token,
- Exchange the user provided access token for an access token for this user to its specific component,
- Retrieve user information,
- Grant access to resources for the identity,
- Pass the token to any other component.

This allows each component to manage the lifecycle of the access token for itself, to renew it using a refresh token of type *offline* when needed for long running task for instance, and this without impacting the other component.

This behaviour is illustrated in Figure 3.

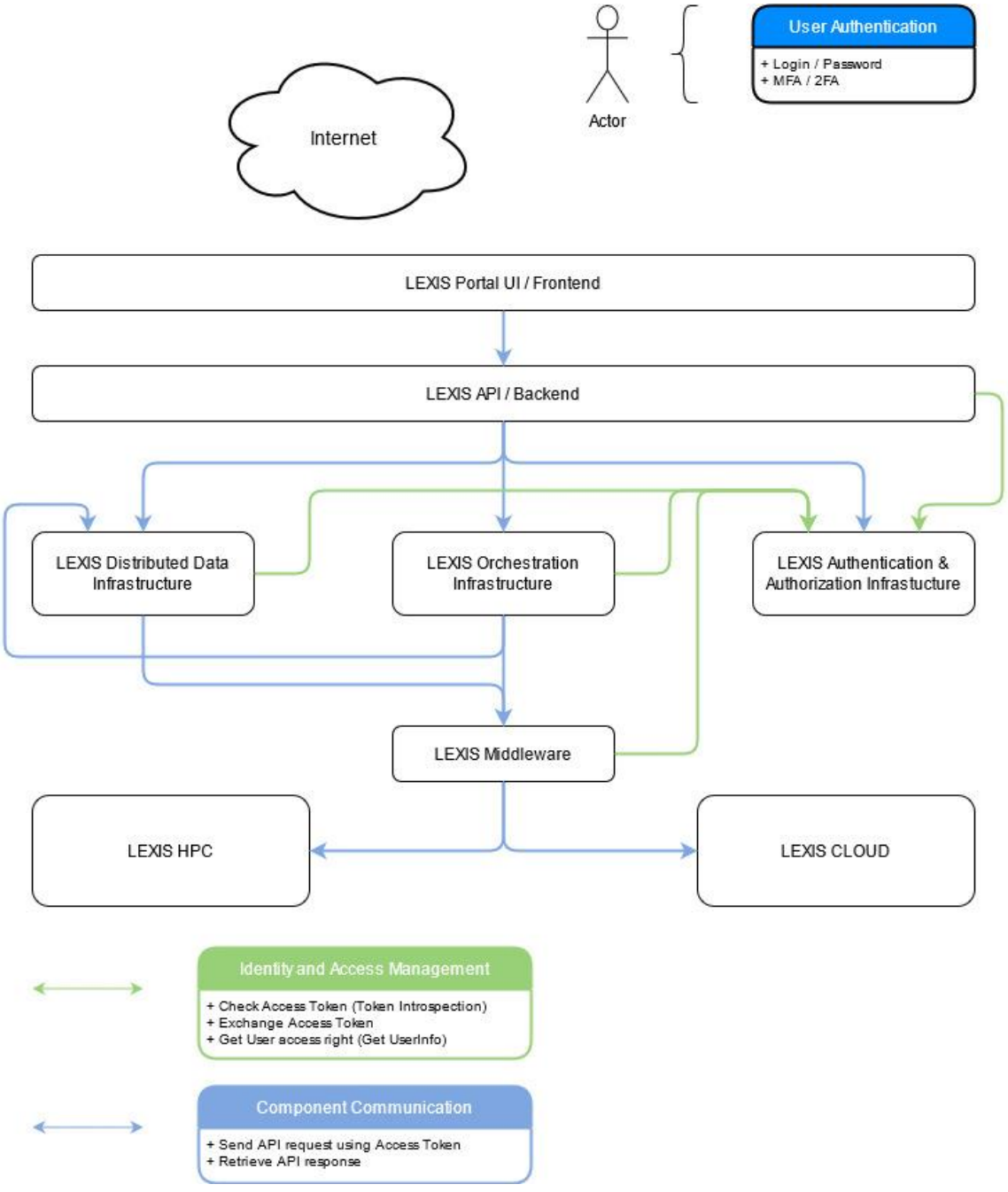

Figure 3 Lifecycle of the access token

Keycloak has also been configured with short life access token and long-time refresh token of type *offline*.

Access token size is kept small and does not contain access information. To avoid any issue with token size, the access information has been added as attribute that can be retrieved during a call to *userinfo* to get user's information.

2.2.1 Keycloak Master & LEXIS AAI Realms

A specific realm named "LEXIS AAI" has been created in Keycloak for the LEXIS project that contains all the identities within LEXIS. This realm is also used as identity provider (IdP) in each supercomputing centre to allow the administrators to access the Keycloak *master* realm and then be able to fully managed Keycloak without any restriction. This "full access" to Keycloak is restricted to each supercomputing centre used for the deployment and not accessible from the internet.

Figure 4 shows the Identity provider configuration when a user wants to login to Keycloak.

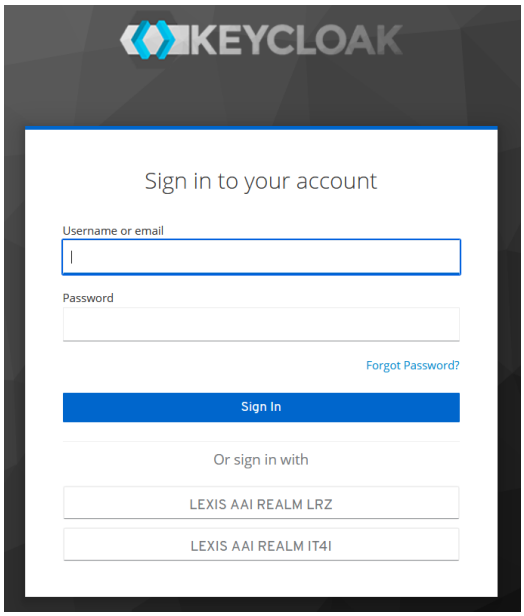

Figure 4 Login to Keycloak

By clicking on a specific Identity Provider, the user is redirected to the LEXIS realm for login as shown in Figure 5.

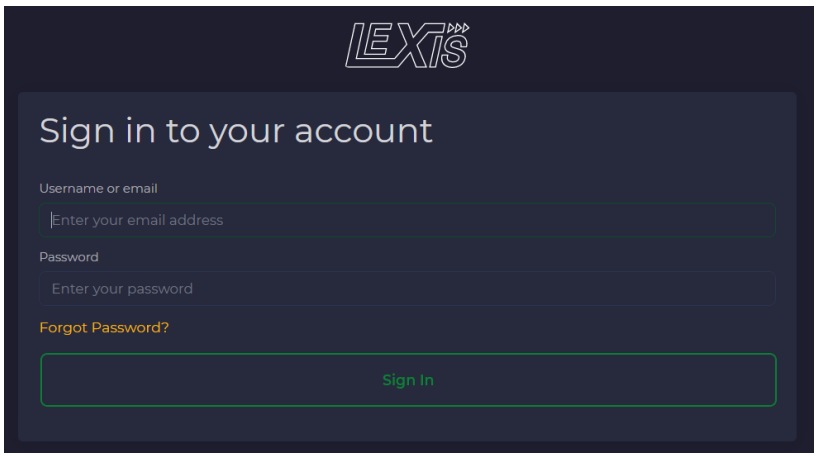

Figure 5 User login - LEXIS realm

The Keycloak internal configuration is as follows:

- 2 Identity Providers configured in Keycloak master realm (one per supercomputing centre), see Figure 6.

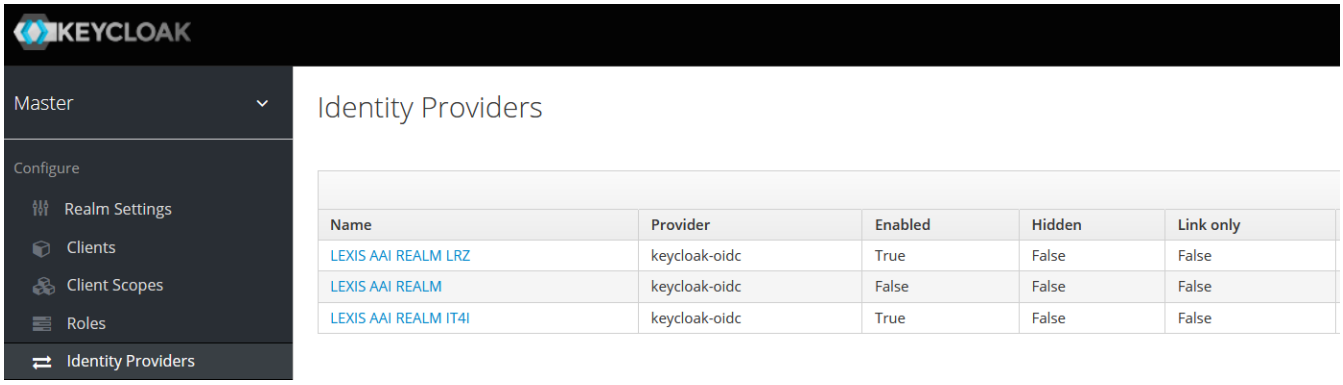

Figure 6 Identity Providers configured in Keycloak master realm

- Each identity provider points to the "LEXI AAI" realm broker in the supercomputing centre, then uses a confidential Keycloak Client named LEXIS_AAI_REALM deployed in LEXIS AAI realm and configured as "confidential", see Figure 7.

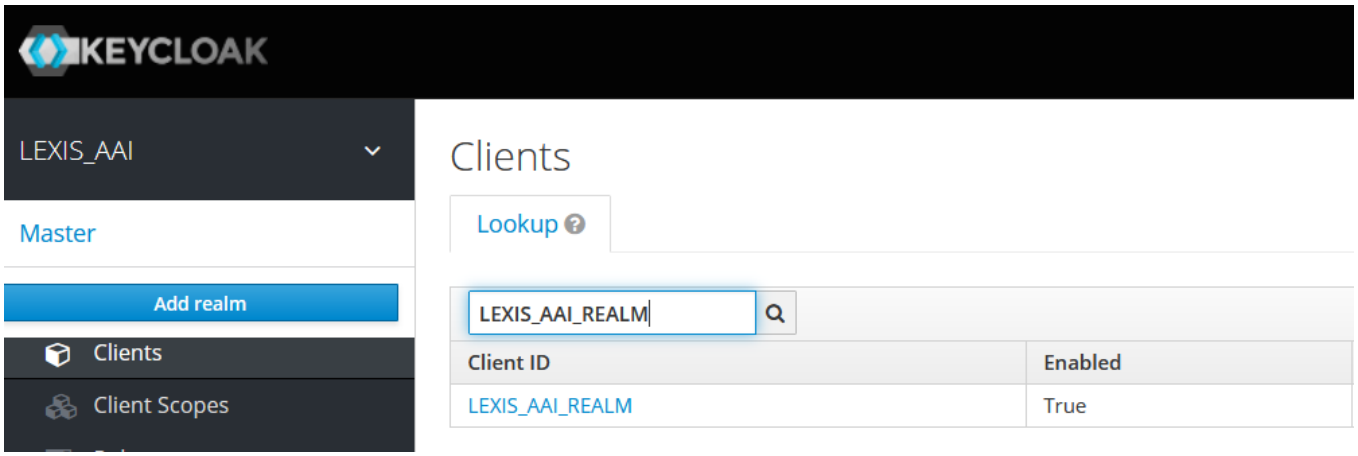

Figure 7 Confidential Keycloak Client deployed in LEXIS AAI realm

- Each identity provider is configured with a Keycloak "mapper" (see Figure 8) to inherit the LEXIS AAI administration role named "LEXIS AAI Admins" setup in the LEXIS AAI realm as a mapped role in a Keycloak Group, see Figure 9.

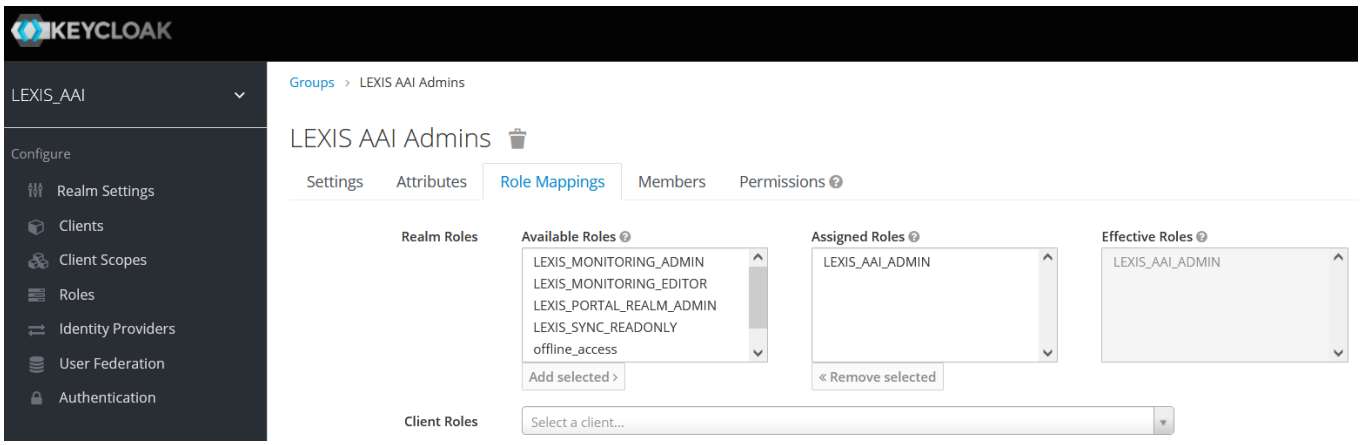

Figure 8 Keycloak role mappings

Roles

Realm Roles | Default Roles

LEXIS_AAI_ | View all roles

Role Name	Composite	Description
LEXIS_AAI_ADMIN	True	Administrator role for LEXIS AAI based on LEXIS_AAI_ADMIN role in LEXIS_AAI_REALM Client

Figure 9 LEXIS AAI realm roles

2.2.2 Keycloak Client for each LEXIS Component

To comply with a zero-trust architecture (ZTA) and follow the least privileges principles, Keycloak has been configured with one Keycloak Client per LEXIS component with "LEXIS_" suffix and a meaningful name as shown in Figure 10.

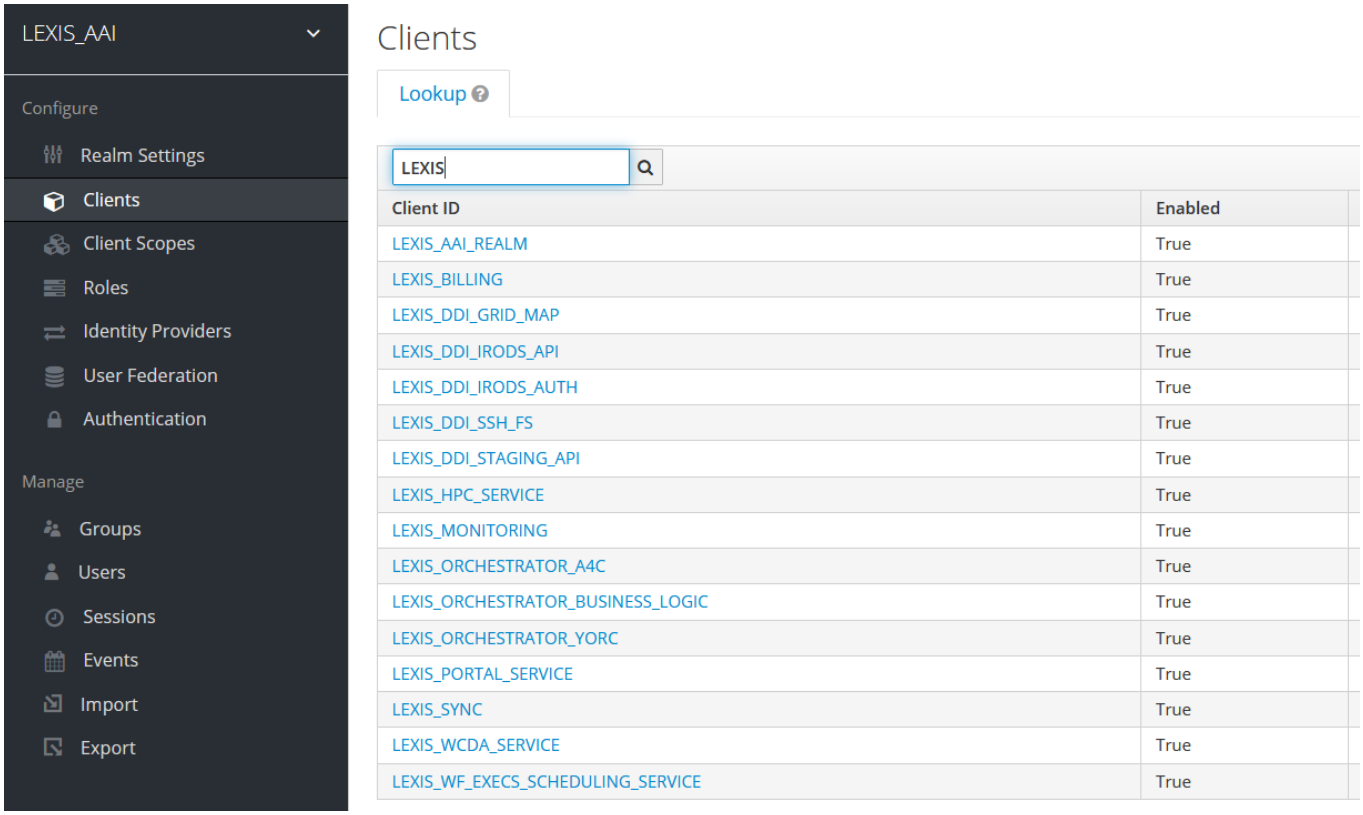

Figure 10 Keycloak Clients for LEXIS components

Each Keycloak Client is configured as "Confidential" with authentication set to "Client Id and Secret". Each component can then access their own Keycloak Client using both the "Client ID" and the "Secret" configured in Keycloak.

As the "exchange token" feature has been enabled on Keycloak, it allows each LEXIS component to exchange a valid token from another component to a specific token for themselves, meaning that they can invalidate, refresh an access token for themselves without impacting the other component.

In a case of having several components using the same Keycloak Client, a component refreshing a token will invalidate all other components' access making them impossible to refresh their token as well. This was not acceptable in the case of LEXIS platform where any component needed to be able to refresh their token as several of them may have long running jobs such as LEXIS DDI and LEXIS Orchestration services.

Please note as well that the LEXIS_AAI_REALM Keycloak Client is used to serve as an identity provider for the Keycloak "Master" realm.

Each Keycloak Client contains mappers to propagate user attributes to the component when it calls *userinfo*, see Figure 11.

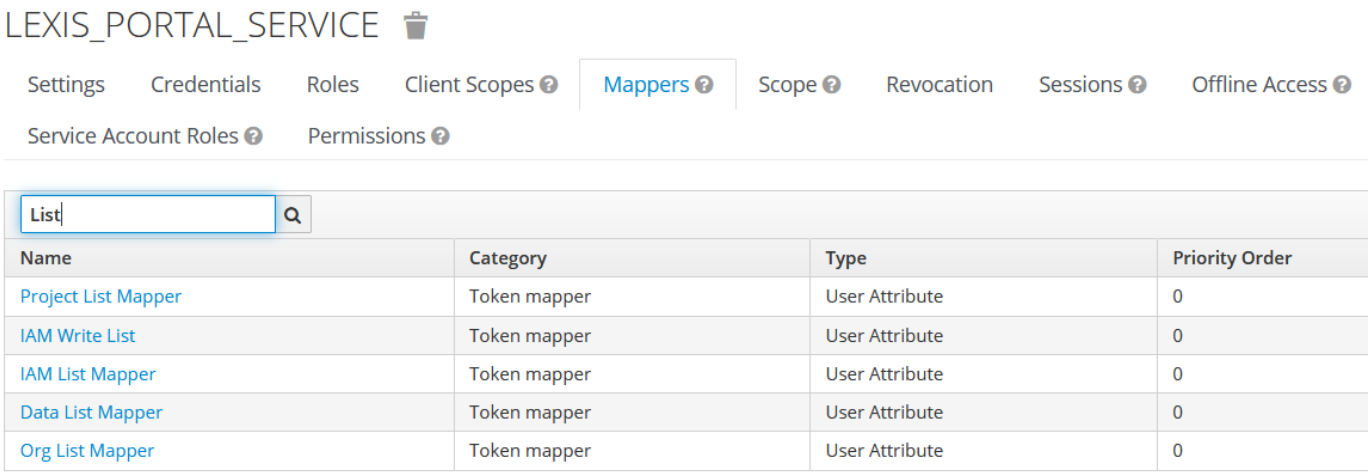

Figure 11 Keycloak Client mappers to propagate user attributes to LEXIS components

As an example, Figure 12 shows the mapper configuration for an organization listing mappers.

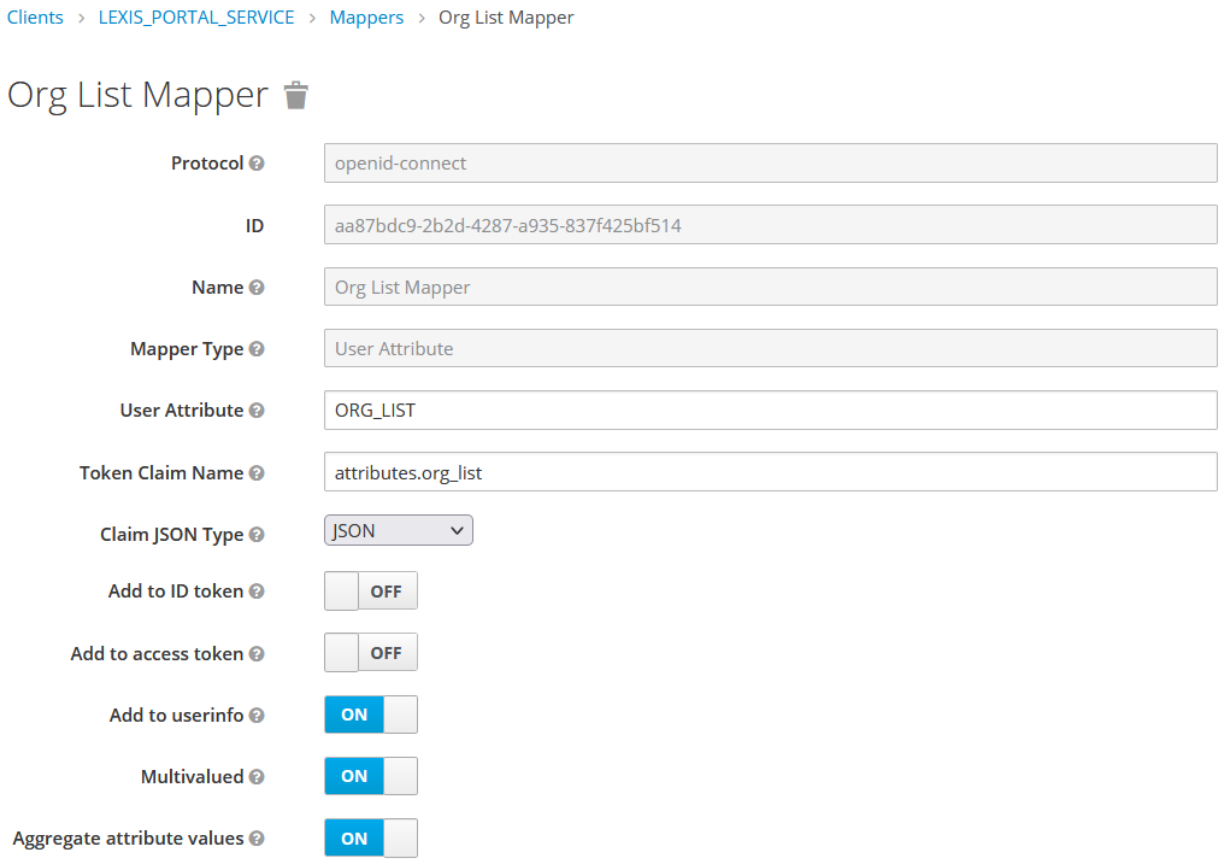

Figure 12 Mapper configuration for an organization listing mappers

Please, note that nothing is exposed in access token and that the "Multivalued" and "Aggregate attribute values" options are selected allowing to specify several organization list permissions for a specific user. Without those options, the permissions will just be overwritten any time something is added and then can reach some inconsistent state.

2.2.3 RBAC implementation using ABAC

As an output of the co-design phase, we decided to grant access to LEXIS Platform implementing attribute-based access control (ABAC) rather than a role-based access control (RBAC), to ease management based on attributes set at a group level.

As a reminder, the ABAC model is based on three permissions as described in D4.1 [1] and a "publishing" permission that had been recently added to be able to make data publicly available:

- List: being able to list resource but not able to get the details or the content,
- Read: being able to read a resource, both details and content,
- Write: being able to execute action (create, update, delete or execute) on a resource,
- Publish: being able to make a resource publicly available.

Keycloak has been configured with specific high-level groups to manage LEXIS platform, organizations and projects as described in Figure 13.

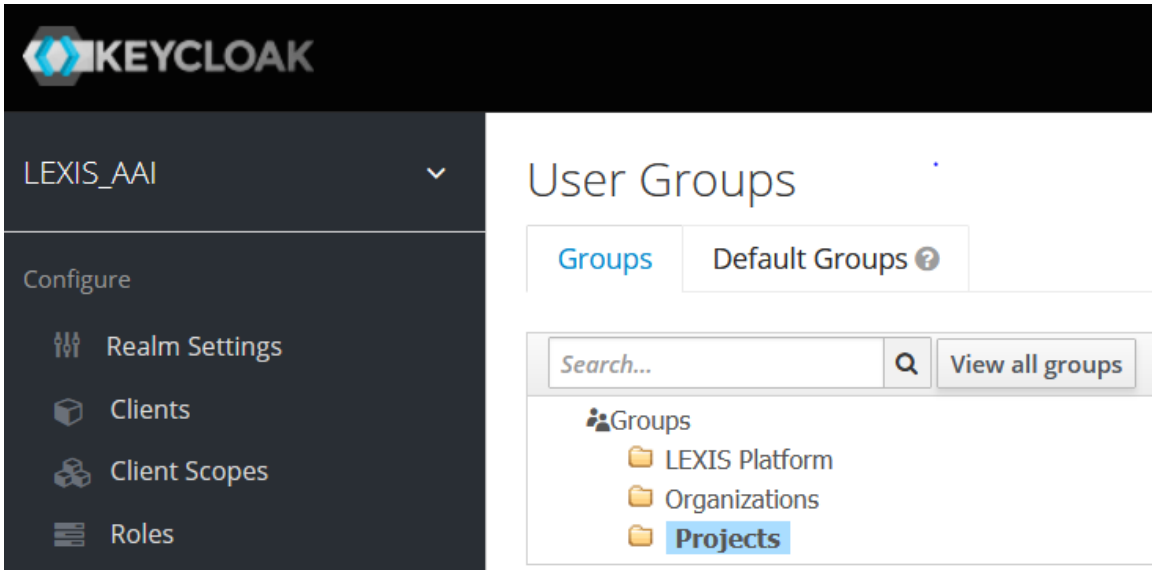

Figure 13 Keycloak high-level groups

The "Organization" group contains all sub-groups to provide access control for organization related access such as Organisation management or Identity and Access management as described in Figure 14:

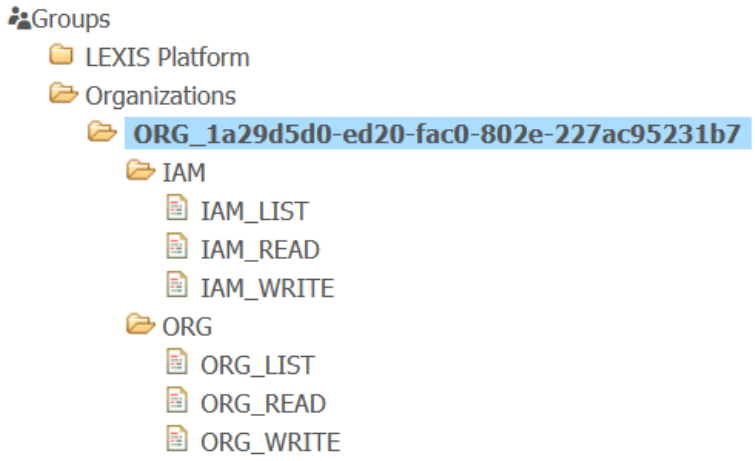

Figure 14 Keycloak: sub-groups in Organization group

In details, the sub-group contains a "key:value" attribute with a JSON formatted value as shown Figure 15:

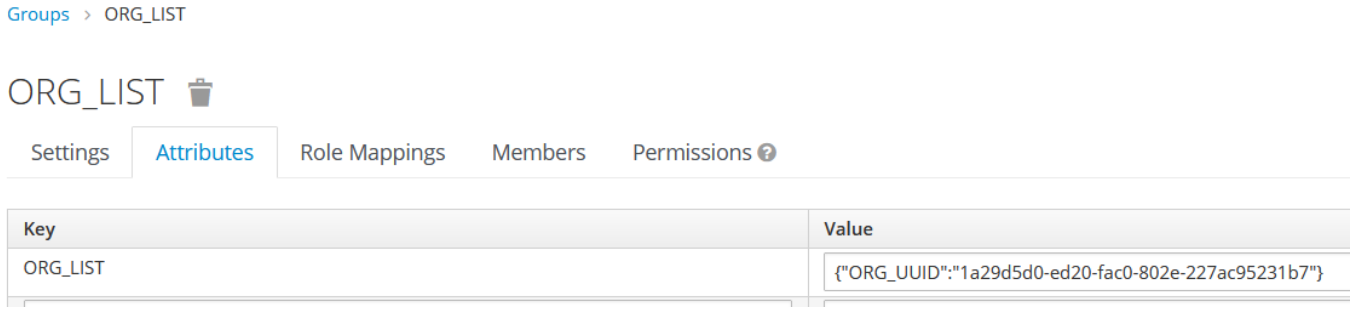

Figure 15 Keycloak: key:value attribute in Organization sub-group

The "Projects" group contains all sub-groups to provide access control for project related access such as computational project management or dataset management as described in Figure 16:

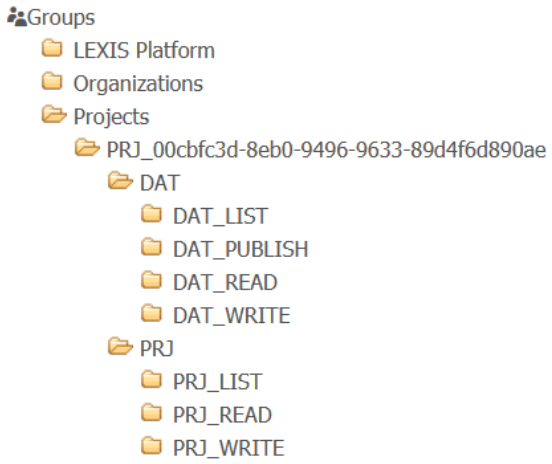

Figure 16 Keycloak: sub-groups in Projects group

In details, the sub-groups contain a "key:value" attribute with a JSON formatted value containing the organization UUID, Project UUID and Project shortname as shown in Figure 17:

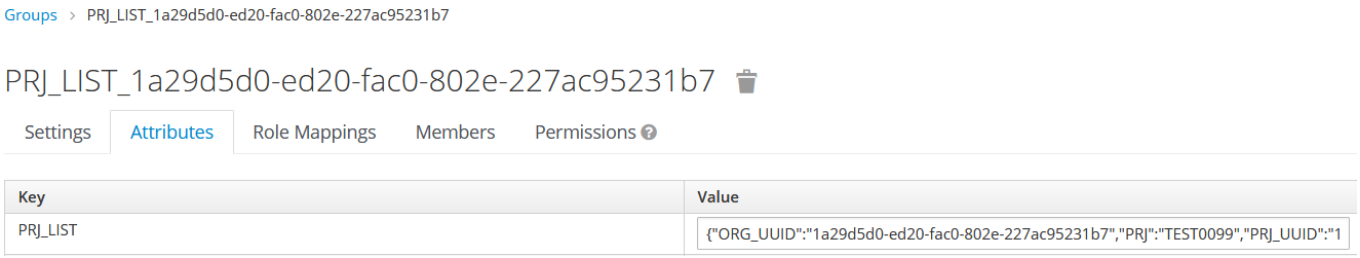

Figure 17 Keycloak: key:value attribute in Projects sub-group

This type is a key:value attribute that allows to restrict the access to a project within an organization, but this also allows to share a project across an organization.

2.2.4 Keycloak Library extension

As described in Section 2.2.3, the RBAC implementation is based on Keycloak's groups. To simplify the life cycle management of these groups, the *user-org* service is using the Keycloak Library extension. This extension aims at providing user-friendly functions needed to implement the RBAC matrix and all the related permissions in Keycloak. These functions group multiple calls to the Keycloak Admin API to create, update or remove the groups and the attributes but also to add or remove users from these groups. For example, to create an organisation in Keycloak (meaning to create all the groups and attributes representing the permissions related to this organisation), the

user-org service just needs to call the *CreateOrganisation* function and all the underlying calls to Keycloak are handled by the library.

Another important point is that the library is responsible for maintaining a consistent and stable state in the event of an error in one of the multiples calls to the Keycloak Admin API. As a result, the latter cleans up Keycloak if any of the calls fail.

3 LEXIS DDI AND EMBEDDED IAM

The LEXIS Distributed Data Infrastructure (DDI) unifies access to LEXIS data across the federated centres. It consists of a distributed Integrated Rule-Oriented Data System (iRODS [3]) and EUDAT-B2SAFE1 [4] installation with multiple so-called iRODS zones, corresponding to the major supercomputing/data centres in LEXIS (see Section 3.1). The iRODS middleware exposes the same structure of files ("Data Objects" in iRODS) and directories ("Collections"), independent of the location or zone through which the system is accessed. Each iRODS zone has its own "filesystem metadata" catalogue ("iCAT") and can operate independently. The iCAT includes access-rights and a user/group database, which one can see as an internal IAM of iRODS (see Section 3.2). By an appropriate configuration, and actually also some coding work, the iRODS system of the LEXIS DDI authenticates users via OpenID Connect (see Section 3.3.1). The identities within the LEXIS IAM/AAI are mapped to iRODS identities in each zone, and access rights are set, so as to best reflect the LEXIS RBAC matrix (see Section 3.3.2). The DDI is (except for strongly restricted gridftp access) exclusively addressed via HTTP-REST APIs, e.g., for data up- and download, data staging requests and data encryption/compression requests. This concept, which is an integral part of LEXIS design and security considerations, implies that the REST APIs comply to OpenID-Connect authentication and the LEXIS RBAC concept as well (see Section 3.3.3).

3.1 REDUNDANT DEPLOYMENT, DDI ZONING AND DATA MIRRORING

As explained above, different geographical sites can be loosely bound in iRODS via a "zone federation" mechanism, which enables transparent data access between the zones, while they are operated independently. Figure 18 gives an overview over the LEXIS iRODS federation concept as of 2020, to which ICHEC is being added. On (transparent) data access, iRODS automatically handles the necessary data transfers with an internal SSL-secured protocol allowing multiple parallel data streams.

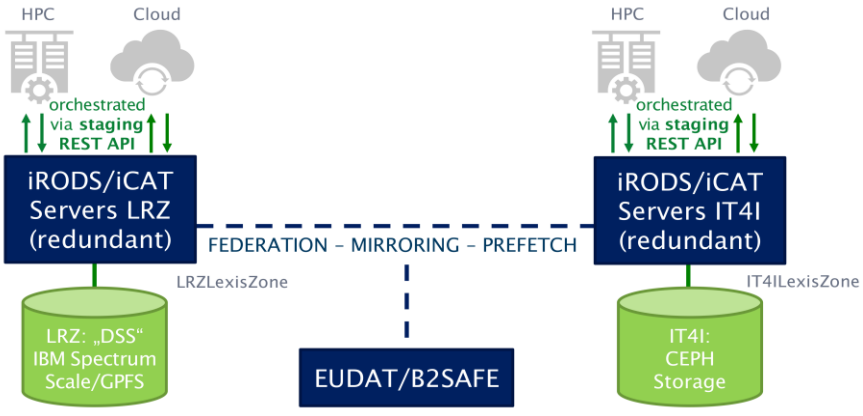

Figure 18 LEXIS DDI federation concept

The zone names (IT4ILexisZone and LRZLexisZone) refer to the two computing/data centres federated in the LEXIS DDI by 2020 (ICHEC is being added). Two main operational back-end storage systems (LRZ's 'Data Science Storage' or 'DSS', and IT4I's Ceph system) are illustrated, as well as the transfer possibilities to Cloud and HPC infrastructure via API calls.

[1] Further EUDAT modules are installed with the DDI (B2HANDLE, B2SHARE) in order to facilitate the assignment of persistent identifiers and the staging of data to HPC systems. These are, however, not relevant in IAM/AAI context.

Each zone in iRODS has a so-called 'iCAT' or 'provider' iRODS server (cf. [3]) which holds the information on the stored data, permissions, local resources and all other necessary zone-specific information in a database (PostgreSQL in our case). In case of major problems in one zone (e.g. long-term power outage), the rest of the DDI infrastructure thus remains operational. To minimize the probability of single-zone failure, in LEXIS each zone itself is set up with a redundant iCAT (cf. [5]), using also a redundant PostgreSQL database backend with a configuration based on repmgr and pgpool (cf. [6]). This per-site setup, following the HAIRS concept [5], is illustrated in Figure 19 and is further discussed in [7, 8]. The system is capable reaching high availability.

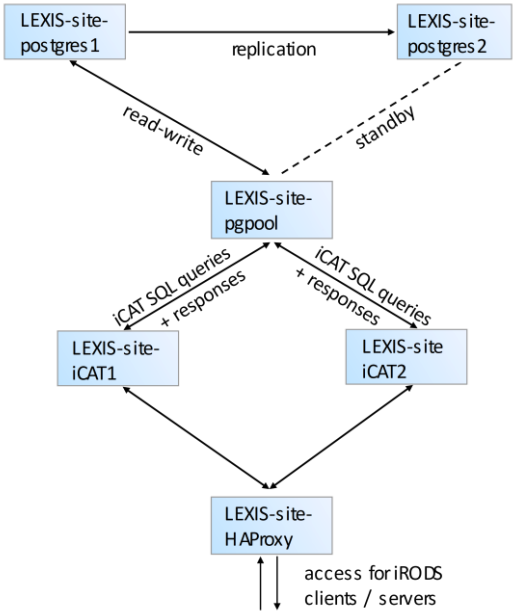

Figure 19 Redundant iRODS-Server setup for one site in the LEXIS DDI. Boxes represent virtual machines.

The LEXIS DDI employs the B2SAFE module of the EUDAT [4] collaboration in order to employ a cross-zone mirroring of data sets if requested by the user. In this case, the mirrored dataset inherits the access rights from the original data set. The geographical data set mirroring, as we offer it, minimizes access times (as it makes cross-zone data transfers unnecessary) and increases data safety and availability at the cost of some additional storage demand.

3.1.1 iRODS IAM

As classical UNIX-like systems, iRODS knows the notion of users and groups. The semantics of the iRODS access management is very similar to that of POSIX file systems, with ACLs regulating the fine-grained rights of these users and groups. In a LEXIS context, we have chosen the groups to correspond to LEXIS Computational Projects, and the users to LEXIS users, in order to warrant a best possible modelling of the LEXIS RBAC concept within the DDI, albeit with some restrictions (see also Section 3.2.2).

iRODS users and groups are stored per zone in the local iCAT, where in LEXIS it is made sure that users and groups exist in all iCAT systems with the same name; inside iRODS these "zone-internal" users and groups are addressed as `<username>#<zonename>` (e.g. `hachinger#LRZLexisZone`) and `<groupname>#<zonename>`. In order to warrant appropriate cross-zone access rights, all zone-specific versions of a LEXIS user (e.g. `hachinger#LRZLexisZone`, `hachinger#IT4ILexisZone`, `hachinger#ICHECLexisZone`) must have the same rights on directories and files. Synchronisation of the access rights via group permissions can be guaranteed when all zone-specific versions of a group are having the same rights for each directory and file.

Via plugins, iRODS allows to delegate the authentication to an external IAM, and in particular allows for an OpenID Connect authentication flow. Problems with the respective plugin when using a Keycloak IAM solution as in LEXIS have been resolved through a conceptual workaround (cf. Section 3.2.1; [9]). Then, only the task remains to set

iRODS rights on data objects and collections (cf. Section 3.2.2) so as to reflect the LEXIS RBAC matrix and allow different levels of data privacy (published data, project-private data, user-private data).

3.2 DDI CONFIGURATION WITH RESPECT TO AUTHENTICATION & AUTHORISATION

In the following sections we give more detail on authentication with the iRODS-OpenID plugin (Section 3.2.1), the identity and access mapping from the LEXIS IAM/AAI to iRODS (Section 3.2.2), and the usage of OpenID Connect by the HTTP-REST APIs through which the LEXIS DDI is accessed and controlled (Section 3.2.3). The reader is thus presented a comprehensive walk-through laying out the relation between the LEXIS AAI and the DDI.

3.2.1 Authenticating LEXIS DDI/iRODS users via OpenID Connect

In 2019, authentication of iRODS users via OpenID Connect, and Keycloak as identity provider, was tested in the LEXIS project. It turned out that the iRODS-OpenID plugin [10] is in principle capable of fulfilling our requirements; however, the relatively long JWT tokens regularly issued by Keycloak cause an overflow in a fixed-size variable within the iRODS authentication system. We worked around this by modifying the iRODS-OpenID plugin architecture, such that tokens are pre-validated and only a token hash transverses the iRODS core [9]. This workaround requires changes to the client programs (e.g., iRODS Python client, icommands), and will be incorporated upstream in iRODS for the version 4.3 release cycle. Our solution was presented at international workshops [11, 9] and the resulting authentication data flow is illustrated in Figure 20.

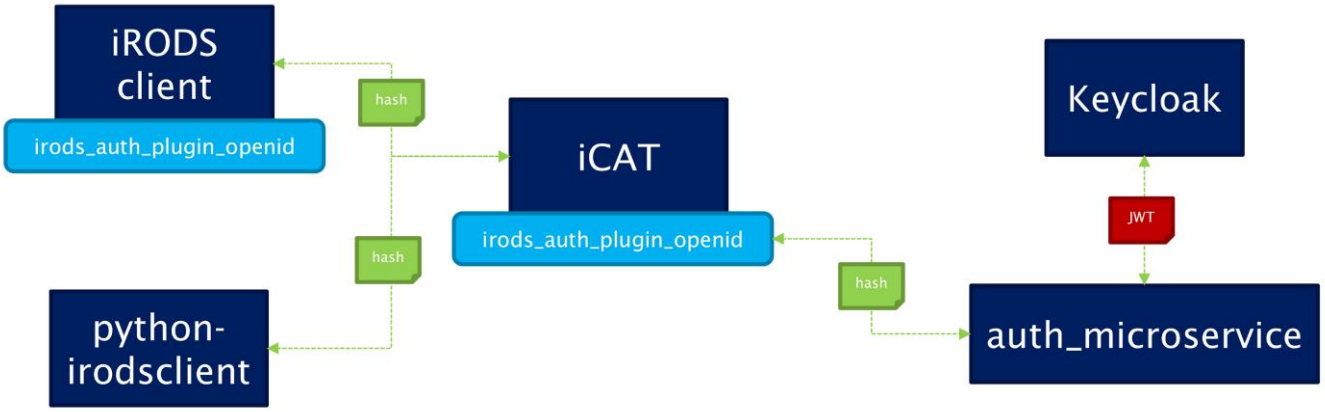

Figure 20. iRODS-OpenID-Connect authentication flow with token pre-validation on client side, hashed token flow, and hash-JWT lookup/validation in auth_microservice.

3.2.2 LEXIS AAI to LEXIS DDI identity and access mapping, RBAC implementation

Besides the iRODS-OpenIDConnect integration, the mapping of access rights according to the LEXIS RBAC matrix to iRODS ACLs has been a non-trivial endeavour. While it was clear relatively early that iRODS groups might best correspond to LEXIS Computational Projects, and iRODS users to LEXIS users, our concept for data object / collection (file / directory) access rights underwent several revisions before reaching maturity (interactions with e.g. EUDAT B2SAFE required modifications).

This has to do with the concept of data scope / data privacy levels in the LEXIS DDI: Data can be “world public“, project private or user private. The directory (or more precisely, iRODS “collection“) structure of our DDI reflects project memberships and privacy levels of data sets. It starts with the iRODS zone (e.g. `/IT4ILexisZone`). Three collections then exist on the next level, which are named “user“, “project“, and “public“, for data sets which can be accessed only by the user, by all the members of a project, or by everybody. At the second level in each of these, collections for each project exist. The third level in “user“ contains a collection for each user to write his data sets

to, while the third level in “project“ and “public“ contain shared datasets. Each data set then is automatically stored in a collection named according to a unique identifier.

Within this directory tree, the ACLs of iRODS have then been set for every collection and data object such that security issues are avoided and users as well as project administrators gain the correct rights (a project administrator can delete project data sets or publish them by moving them to the public collection hierarchy). iRODS provides the access levels “own“, “write“ and “read“. “own“ is used to allow the user to modify the permissions for themselves or other users. “write“ and “read“ have the usual meaning, with “read“ including listing (therefore the RBAC distinction between list and read is implemented at the API level). In addition, iRODS collections can have an inheritance flag enabled, which makes subcollections and data objects in them inherit the parent rights when created.

The following users and groups are created:

- For each Lexis user <U> and zones <Z> where they are authorized: the iRODS users <U>#<Z>LexisZone,
- For each Lexis project <P>: the iRODS groups <P> and <P>_mgr,
- A group lexis_group providing read access to public datasets,
- A group rodsadmin providing access to administrative tasks,
- Groups corresponding to scope-specific LEXIS roles with an appropriate naming, e.g. lex_adm, lex_sup.

The users <U>#<Z>LexisZone belong to the lexis_group and to any project <P> they have access to. Users with administrator rights to a project P are also added to group <P>_mgr.

On user creation or modification, the group is removed from all groups and added to the lexis_group and the project groups they should belong to; and to additional scope-specific roles as needed.

For the directory structure, we are maintaining confidentiality by hashing the project names with the md5 algorithm and making datasets names UUIDs. A project P receives a directory proj<MD5 P>. This ensures that EUDAT-exported metadata containing paths do not expose the project name.

We set the directory structure and rights as follows:

- /<Z>LexisZone: No access, Inheritance disabled
- 1st level /<Z>LexisZone/project, /<Z>LexisZone/user, /<Z>LexisZone/public: lexis_group: read, rods, rodsadmin: own, Inheritance disabled
- 2nd level /<Z>LexisZone/user/<P>: rods, rodsadmin: own, Inheritance disabled
- 2nd level /<Z>LexisZone/project/<P>: rods, rodsadmin, <P>, <P>_mgr: own, Inheritance enabled
- 2nd level /<Z>LexisZone/public/<P>: rods: own, lexis_group: read, <P>: read, <P>_mgr: own, Inheritance enabled
- 3rd level /<Z>LexisZone/user/<P>/<U>: rods, rodsadmin: own, <U>#<Z>LexisZone: own, Inheritance enabled
- Dataset directories inherit permissions from the parent directory.

In addition, after publication of a dataset (moving it within the /public/ hierarchy, the dataset is frozen (no further modification is possible); therefore, permissions are recursively set to none for <P> and <P>_mgr.

Permissions are also given and removed to the lex_sup group (to specific files) when support tasks regarding those files are requested, and when the request is finalized.

Some features of the LEXIS RBAC concept (e.g., permission to list vs read datasets) cannot be modelled with the scheme above or any minor extensions of it. Therefore, the APIs for accessing the LEXIS DDI are programmed such as to directly reject unauthorized requests of this kind and not let them arrive at the iRODS layer (Section 32.3).

The usage of GridFTP access for high-speed upload and download of datasets for power-users bypasses the LEXIS APIs, so it may only be enabled for users with write permissions to all their datasets, to avoid permission escalation.

3.2.3 OpenID Connect usage and RBAC implementation within the LEXIS DDI APIs

The HTTP-REST APIs of the LEXIS DDI (cf. [12, 13]) receive tokens with the HTTPS requests other LEXIS systems (orchestrator, portal) send them on behalf of the user. Following the zero-trust concept, the APIs check these tokens against the LEXIS IAM system and exchange them by a service-specific offline token and a refresh token (the refresh token is used for long-term, multi-step operations; a new token is requested for each step to ensure the user remains authenticated). In the offline token, not only the user data, but also the relevant roles of the user are encoded. Thus, parts of the LEXIS RBAC scheme which cannot be directly implemented via iRODS rights are implemented within the APIs. The API endpoint workflow is thus:

- Verify validity of token, returning 401 Unauthorized if token is not valid,
- Retrieve refresh and offline tokens,
- Check the permissions needed according to the specific endpoint and parameters against the offline token data; return 403 Forbidden if the user is not authorized,
- Perform the task on behalf of the user and return appropriate response.

The RBAC roles relevant for DDI are the LEXIS data manager and the LEXIS end user. The LEXIS end user can create, modify, download and delete datasets, list their files, read and modify metadata and perform metadata search. With respect to staging, he can perform staging tasks and export data via SSHFS or NFS for usage in cloud services with LEXIS. The data manager has additional permissions, such as creation and deletion of iRODS users and projects, managing users within projects, and setting additional roles for users. Keycloak permissions to list, read and write datasets within a project are also checked at the corresponding API endpoints.

4 LEXIS MIDDLEWARE AND HPC, CLOUD IDENTITIES

4.1 HEAPPE MIDDLEWARE

HEAppE Middleware[2] is IT4Innovations' an in-house implementation of HPC-as-a-Service concept. HPC-as-a-Service is a well-known term in the area of high-performance computing. It enables users to access an HPC infrastructure without the need to buy and manage their own physical servers or data centre infrastructure. This approach further lowers the entry barrier for users who are interested in utilizing massive parallel computers but often do not have the necessary level of expertise in this area.

HEAppE Middleware manages and provides information about submitting and running jobs and their data between the client application and the HPC infrastructure. HEAppE can submit required computation or simulation on HPC infrastructure, monitor the progress and notify the user when necessary. It provides necessary functions for job management, monitoring and reporting, user authentication and authorization, file transfer, encryption, and various notification mechanisms.

HEAppE Middleware main features:

- Multi-platform .NET Core version,
- OpenAPI REST API.
- Dockerized deployment and management,
- Updated PBS and Slurm workload manager adapter,
- SSH Agent support,
- Multiple tasks within single computational jobs,
- Job arrays support and job dependency support,
- Extremely long running job support,
- OpenID and OpenStack authentication support.

[2] HEAppE Middleware: https://heappe.eu/

4.1.1 HEAppE Identities

HEAppE Middleware utilizes two types of identities: external user accounts and internal infrastructure accounts. Internal infrastructure accounts will be described in the following section.

External user accounts or so-called general HEAppE identity is used to access HEAppE's REST API main functions upon authentication. As HEAppE provides remote access to HPC infrastructure, these user identities represent individual external users or companies that does not have a direct access to the HPC infrastructure and are only able to invoke the functions provided by the HEAppE's REST API.

HEAppE identities information is stored within HEAppE's internal relational database and is managed by a HEAppE's admin team. One of the core functions of the HEAppE Middleware is the mapping functionally between the external user identities and internal infrastructure ones. Utilizing this approach, even the users without direct access to the HPC infrastructure can remotely access it, submit the computation, manage their jobs and monitor their progress. This mapping functionality is described in more detail in Section 4.1.3.

4.1.2 HPC and Cloud Identities

HPC identities are represented by internal HPC cluster accounts associated with a specific computational project in an HPC centre. Primary Investigator (PI) of a computational project must get authorization by the *Allocation Commitee* to access and use HPC centre clusters. The PI can then grant access to other members of the project to use the resources. For security reasons, the created cluster accounts are then used in a specific HEAppE instance via the SSH Agent instance running in a standalone Docker container thus separated from the core HEAppE Middleware instance and its internal database.

There is a slightly different approach for the Cloud identities than for the HPC infrastructure where the HEAppE Middleware acts as a complete wrapper for the entire HPC related functionality. The Cloud identities are handled by the Cloud infrastructure administrators and HEAppE wraps only the authentication part to the Cloud infrastructure in its REST API as explained in the next chapter.

4.1.3 LEXIS AAI to HPC and Cloud identities and access mapping

See Figure 21 for a view of the HPC and Cloud identity mapping mechanism overview. When the LEXIS user or some part of the LEXIS platform (e.g., DAM/Ystia orchestrator) wants to use the HPC/Cloud infrastructure, first the user needs to authenticate via the Keycloak instance. Upon successful authentication, the user is provided with a Keycloak token that is a mandatory parameter of mostly all the HEAppE's REST API endpoints.

In the HPC context, this token is required to create/submit/monitor/manage the compute jobs, upload/download the jobs' data or gather the information about the infrastructure itself. During the HEAppE endpoint execution, the Keycloak token is verified against the appropriate Keycloak Client (to ensure access token validity) and the access is granted according to the information (contained in the user attributes) retrieved from the Keycloak Client.

In the Cloud context, the HEAppE serves only as an AAI middleware. Via the HEAppE's REST API, the user is able to use the Keycloak token to authenticate within Cloud environment via OpenStack application credentials and is provided with OpenStack application credentials. This application credentials are transformed to a Keystone token by the underlying application (e.g., DAM). Keystone token can then be directly used within the OpenStack environment without further HEAppE's involvement.

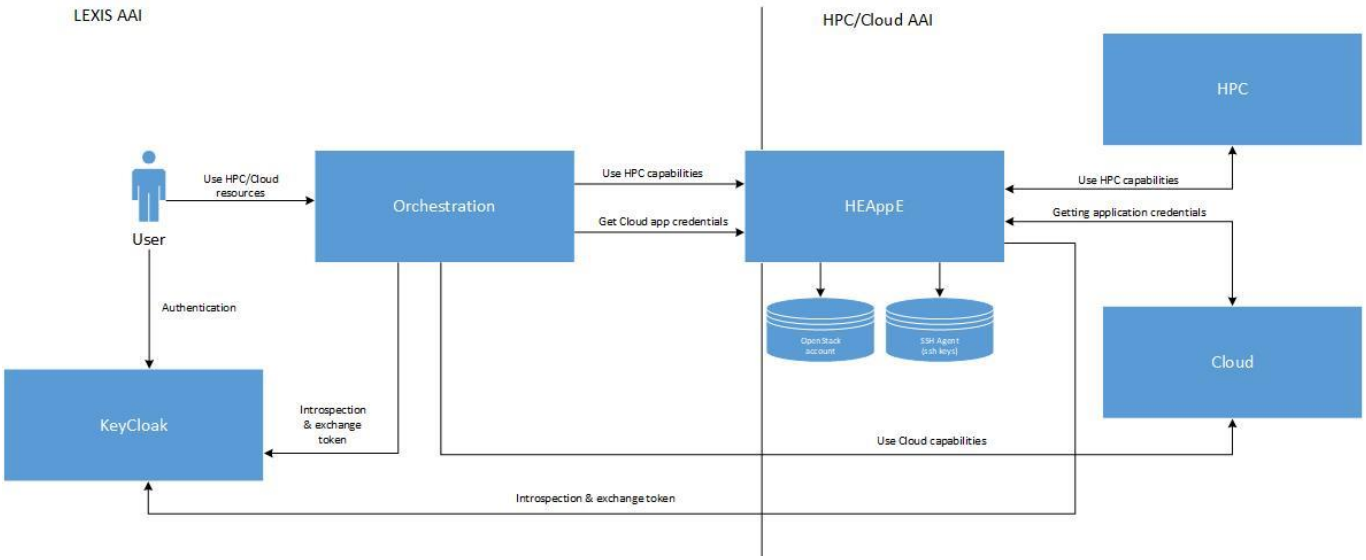

Figure 21 HPC and Cloud identity mapping

4.1.4 OpenIDC for LEXIS AAI

Each instance of HEAppE Middleware connects to the LEXIS platform's AAI via a common LEXIS AAI HA proxy. HEAppE Middleware integrates Zero Trust Architecture, which means that HEAppE always checks the validity of the OpenID token provided by the applications above. Each HEAppE Middleware instance is deployed for a specific computational project in a specific centre and LEXIS Project UUID and Project name mapped in LEXIS AAI. This set-up information varies for each individual HEAppE Middleware instance.

4.2 CONFIGURATION

Dockerization is used for the deployment of the HEAppE Middleware instances. Docker containers coordinated using docker-compose which is providing containers dependency, simplification, etc.

The HEAppE Middleware docker-compose consists of these docker containers:

- HEAppE middleware (API),
- MS-SQL Database,
- SSH Agent.

Prerequisites:

- Two or more functional/service accounts in the same group,
- Access to GitLab repository with HPC cluster scripts,
- Access to GitLab repository with HEAppE Middleware,
- Virtual machine minimum requirements:
    - 2 CPUs,
    - 4 GB RAM,
    - 50 GB HDD,
    - Unix based OS,
    - Ports
        - Ingress - 22, 5000, 6000, 80/443,
        - Outgress - 22.
    - SW

    - Git (recommended),
    - Network-utils (recommended),
    - Epel-release (recommended),
    - Wget (recommended),
    - Nginx (recommended in case of usage proxy for HTTPS),
    - Docker,
    - Docker-compose.
- HPC cluster scripts preparation
    - HEAppE's service scripts (used to create job's folder, copy data to/from temporary storage, etc.) are deployed on an HPC infrastructure under the functional/service accounts

Preparation of configuration files:

- `.env`
    - Exposed ports and disk volumes mapping,
- `appsetings.json`
    - Logging section,
    - Application API section,
    - Database connection section,
    - IP rate limitation section,
    - IP rate limit policies section,
    - KeyCloak configuration section,
    - OpenStack configuration section.
- `seed.json`
    - `AdaptorUsers` - external user accounts, which are mapped to HPC users
    - `AdaptorUserGroups` - groups, which are assigned to external users
    - `AdaptorUserUserGroups` - binding table between AdaptorUsers and AdaptorUserGroups
    - `AdaptorUserRoles` - roles in HEAppE (not currently used)
    - `Clusters` - HPC cluster specification
    - `ClusterNodeTypes` - queues for each HPC cluster
    - Cluster authentication credentials - functional accounts
    - `CommandTemplates` - template specifications for job execution
    - `CommandTemplateParameter` - command template parameters
    - `JobTemplates` - default HPC job values
    - `TaskTemplates` - default HPC task values
    - `OpenStackInstances` - OpenStack instance endpoint
    - `OpenStackAuthenticationCredentials` - credentials for OpenStack

5 CONCLUSION

The detailed description of the centralized LEXIS authentication and authorization infrastructure inside the LEXIS platform explained the need to have two models for onboarding and integrating a new supercomputing centre. As a matter of fact, the LEXIS AAI relying on Keycloak IAM open source solution is accessible from the internet as well as from any other supercomputing centre, making it easier to only deploy the required components for providing the computational (mapping supercomputing user with HEAppE Middleware) or data service (mapping system user with iRODS).

Both the compute and data services rely on their own identity and access management system that cannot be easily accessed from any other component. As such, LEXIS AAI is acting as the main central point to coordinate and synchronize the identity and access related requests. It represents the source of truth for all other components and identity providers.

REFERENCES

[1] LEXIS Deliverable, *D4.1 Analysis of Mechanism for Securing Federate Infrastructure.*

[2] LEXIS Deliverable, *D4.5 Definition of Mechanisms for Securing Federated Infrastructures.*

[3] H. Xu, T. Russell, J. Coposky, A. Rajasekar, R. Moore, A. de Torcy, M. Wan, W. Shroeder and S. Chen, iRODS primer 2: Integrated Rule-Oriented Data System, Williston, VT: Morgan & Claypool Publishers, 2017.

[4] "EUDAT Collaborative Data Infrastructure: B2SAFE-EUDAT," 2020. [Online]. Available: https://www.eudat.eu/services/b2safe. [Accessed 6 Nov 2020].

[5] Y. Kawai and A. Hasan, "High-Availability iRODS System (HAIRS)," in *Proceedings of the iRODS User Group Meeting 2010: Policy-Based Data Management, Sharing and Preservation*, NC, 2010.

[6] J. Depuydt, "Setup a redundant postgresql database with repmgr and pgpool | jensd's i/o buffer," 2015. [Online]. Available: http://jensd.be/591/linux/setup-a-redundant-postgresql-database-with-repmgr-and-pgpool. [Accessed 6 Nov 2020].

[7] J. James, "Configuring iRODS for High Availability," iRODS, 7 July 2015. [Online]. Available: https://irods.org/2015/07/configuring-irods-for-high-availability/. [Accessed April 2020].

[8] J. James, "UGM 2016 - High Availability with iRODS," June 2016. [Online]. Available: https://slides.com/justinjames-1/ugm2016-high-availability#/.

[9] R. J. García-Hernández and M. Golasowski, "Supporting Keycloak in iRODS systems with OpenID authentication," in *CS3 2020 - Workshop on Cloud Storage Synchronization and Sharing Services*, 2020.

[10] K. Ferriter, "OpenID Connect Authentification in iRODS," in *iRODS User Group Meeting*, Durham, 2018.

[11] M. Golasowski, M. Hayek and R. J. García-Hernández, "A transnational data system for HPC/Cloud-Computing Workflows based on iRODS/EUDAT," in *iRODS virtual user group meeting*, 2021.

[12] "LEXIS Developer Guide," September 2020. [Online]. Available: https://docs.lexis.tech/_pages/portal/developer_guide.html.

[13] LEXIS Deliverable, *D3.2 Mid-Term infrastructure.*

[14] A. Rajasekar, R. Moore and et al., "iRODS primer: integrated rule-oriented data system," *Synthesis Lectures on Information Concepts, Retrieval, and Services,* vol. 2, no. 1, pp. 1-143, 2010.