

OGC Testbed-15
Federated Clouds Security Engineering Report

Table of Contents

1. Subject	4
2. Executive Summary	5
2.1. Document contributor contact points	6
2.2. Foreword	6
3. References	7
4. Terms and definitions	8
4.1. Abbreviated terms	8
5. Overview	10
6. An Introduction to Federation	11
6.1. What is a federation?	11
6.2. Security and Federation	12
6.2.1. Identity Federation - Authentication	13
6.2.2. Resource Federation - Authorization	14
6.2.3. Data Federations	16
6.2.4. Service Federations	16
6.3. Deployment Models and Governance	17
6.3.1. Resource Owners	17
6.3.2. Onboarding Requirements	17
6.3.3. Federation Maintenance Maturity	18
7. Federation Management Architecture	19
7.1. Introduction	19
7.2. Deployment models	19
7.2.1. Topology	20
7.2.2. Relationship with Sites	20
7.2.3. Federation Manager Hierarchy	20
7.2.4. Deliverable deployments	21
7.3. Federation Manager to Service Provider Interactions	21
7.3.1. Federation Membership Management	21
7.3.2. Federation Resource Management	22
7.3.3. Federation Policy Management	23
7.4. Federation Manager to Federation Manager Interactions	24
7.4.1. Federation Portability and Interoperability	24
8. Federation Manager #1	27
8.1. Design Overview	27
8.2. Functionality	28
8.2.1. Federation Authentication: Membership Management	28
8.2.2. Federation Authorization: Resource and Policy Management	29
8.2.3. Inter-Federation	32

8.3. Analysis and Future Work	34
9. Federation Manager #2	36
9.1. Introduction	36
9.2. Federation Definition	36
9.3. rasdaman Architecture Overview	36
9.4. rasdaman Federation Overview	38
9.5. Access Control	39
9.5.1. Authentication	40
9.5.2. Trust	41
9.5.3. Authorization	41
9.6. rasdaman as Federation Manager #2 in Testbed-15	43
9.7. Analysis and Future Work	44
10. Technology Integration Experiments	46
10.1. Components	46
10.2. Use Cases	46
10.3. Pairings	48
10.4. Experiments	49
10.4.1. FM#1 propagates authentication information to FM#2	49
10.4.2. FM#1 consumes service metadata from FM#2 and populates its resource catalogue ...	49
10.4.3. FM#2 propagates authentication information to FM#1	49
10.4.4. FM#2 administrator updates user privileges on FM#1 membership service	49
10.4.5. FM#1 administrator updates user privileges on FM#2 membership service	49
10.4.6. One FM systematically updates user privileges on other FM membership service	50
10.5. Summary	50
Appendix A: Revision History	51
Appendix B: Bibliography	52

Publication Date: 2019-12-20

Approval Date: 2019-11-22

Submission Date: 2019-10-04

Reference number of this document: OGC 19-024r1

Reference URL for this document: <http://www.opengis.net/doc/PER/t15-D019>

Category: OGC Public Engineering Report

Editor: Hector Rodriguez

Title: OGC Testbed-15: Federated Clouds Security Engineering Report

OGC Public Engineering Report

COPYRIGHT

Copyright © 2019 Open Geospatial Consortium. To obtain additional rights of use, visit <http://www.opengeospatial.org/>

WARNING

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Public Engineering Report should not be referenced as required or mandatory technology in procurements. However, the discussions in this document could very well lead to the definition of an OGC Standard.

LICENSE AGREEMENT

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD. THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER'S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to

indicate compliance with any LICENSOR standards or specifications.

This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

None of the Intellectual Property or underlying information or technology may be downloaded or otherwise exported or reexported in violation of U.S. export laws and regulations. In addition, you are responsible for complying with any local laws in your jurisdiction which may impact your right to import, export or use the Intellectual Property, and you represent that you have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

Chapter 1. Subject

This OGC Testbed-15 Engineering Report (ER) documents the concept for different types of federation through the lens of security. The primary focus of the work documented in this ER is on analyzing how federated membership, resource and access policy management can be provided within a security environment, while also providing portability and interoperability to all stakeholders.

In the Testbed, a practical approach for providing this functionality was exercised and documented for two administrative domains: One based on a centralized entity (Federation Manager) and the other showcasing a distributed architecture.

Chapter 2. Executive Summary

This ER documents the analysis of both the centralized entity and distributed architecture approaches through the lens of security. The work presented in the ER also took into account how governance can be affected by design choices. The ER also introduces the concept of a Federation Manager (FM) and tries to establish a clear list of functionality aspects necessary to manage and use a Federation.

Therefore, the requirements addressed by this ER are:

- Provide an introduction to the concepts of Federation and Federated Clouds.
- Explain what a Federation Manager is, which are the different deployment models, and their impact in governance models.
- Document an in-depth review of all key functional aspects for a Federation Manager, such as:
 - Membership Management;
 - Policy Management;
 - Portability and Interoperability.
- Provide a complete analysis of the Federation Managers deployed during this Testbed-15 activity, and how to tackle each of the functionality challenges previously mentioned.

To better understand interoperability constraints and address them by proposing architecture and security design options, this ER has two main associated components within Testbed-15: Federation Manager #1 and Federation Manager #2. Both of these logical entities aim at providing the same functionality aspects to Federation members and aim to interact with each other seamlessly. Nevertheless, it is important to take into account that both Federation Managers oppose each other in their approach to fulfill the requirements mentioned above.

Following the strategy defined in the Testbed-15 roadmap, a "one-way" federation was established where FM#2 connects with the authentication functionality provided by FM#1. The testbed also sought to identify an approach to solve the differences between the two techniques in order to facilitate future interoperability of federations.

The results of the Technology Integration Experiment (TIE) pairing between FM#1 and FM#2 indicated that both architectures lead to different technical consequences which have been investigated and described in this engineering report. As such, the engineering report forms a suitable basis for further investigation on the interoperability between and across federations. As for recommendations, this engineering report puts forward the following key recommendations and future work areas:

- Analyze security gateway solutions for OGC Web Services. This ER identifies a necessity to create Security Gateway solutions that ease the effort in closing the gap between state-of-the-art technologies for Federated Clouds, and OGC Web Services.
- Provide a Federation Management REST API. Both FM#1 and FM#2 have run into interoperability issues that could be solved with API endpoints exposed by both components, providing an abstraction layer on top of their already existing solution.

- Secure Workflow execution across domains. Building upon the work performed on Testbed-15, the execution of secured workflows across security domains could already be demonstrated (using Federation Management components)
- Accounting and Billing. Based on the work carried out during this Testbed, it is necessary to further analyze how Accounting as a Service can be delivered by a Federation (either as a brokerage component within a Federation Manager or as a standalone service-side component).
- Definition of governance rules for federations, including prefabricated specializations for standard situations such as: scientific open services; scientific open services with embargos; commercial services with paid access; tightly protected services and private federations.

2.1. Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Contacts

Name	Organization	Role
Hector Rodriguez	Deimos Space	Editor
Peter Baumann	rasdaman	Contributor

2.2. Foreword

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

Chapter 3. References

The following normative documents are referenced in this document.

- **OGC: OGC 06-121r9, OGC® Web Services Common Standard** [https://portal.opengeospatial.org/files/?artifact_id=38867&version=2]
- **IETF: RFC 6749 - The OAuth 2.0 Authorization Framework** [<https://tools.ietf.org/html/rfc6749>]
- **IETF: RFC 6750 - The OAuth 2.0 Authorization Framework: Bearer Token Usage** [<https://tools.ietf.org/html/rfc6750>]
- **IETF: RFC 7662 - OAuth 2.0 Token Introspection** [<https://tools.ietf.org/html/rfc7662>]
- **OIDF: OpenID Connect Core 1.0** [https://openid.net/specs/openid-connect-core-1_0.html]
- **IETF: Internet-Draft - User-Managed Access (UMA) 2.0 Grant for OAuth 2.0 Authorization** [<https://tools.ietf.org/html/draft-maler-oauth-umagrants-00>]
- **IETF: Internet-Draft - User-Managed Access (UMA) 2.0 Grant for OAuth 2.0 Authorization** [<https://tools.ietf.org/html/draft-maler-oauth-umafedauthz-00>]
- **IETF: RFC 7644 - System for Cross-domain Identity Management: Protocol** [<https://tools.ietf.org/html/rfc7644>]
- **OGC: OGC 09-146r6, Coverage Implementation Schema (CIS), version 1.1** [<http://docs.opengeospatial.org/is/09-146r6/09-146r6.html>]
- **OGC: OGC 17-089r1, OGC® Web Coverage Service (WCS) Interface Standard - Core, version 2.1** [<http://docs.opengeospatial.org/is/17-089r1/17-089r1.html>]
- **OGC: OGC 08-068r2, OpenGIS Web Coverage Processing Service (WCPS) Language Interface Standard** [http://portal.opengeospatial.org/files/?artifact_id=32319]

Chapter 4. Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard [OGC 06-121r9](https://portal.openeospatial.org/files/?artifact_id=38867&version=2) [https://portal.openeospatial.org/files/?artifact_id=38867&version=2] shall apply. In addition, the following terms and definitions apply.

- Security Environment

A system that securely manages end-user information for the purpose of providing Identity Management and Access Control. These capabilities are usually achieved by using cryptographic methods, secure network design, and observing data protection regulations.

- Administrative Domain

A system that augments a Security Environment by defining a desired set of policies and governance for managing users and resources. This includes defining an authoritative source for policy concerning resource discovery and use.

- Federated Environment

The creation and management of a Virtual Administrative Domain whereby the same kind of policies and governance can be used to manage users and resources within the VAD that are, in fact, coming from an arbitrary number of non-federated Administrative Domains. This depends on a Federated Security Environment.

- Inter-Federation

An established relationship between two or more already existing Federations. This relationship requires a set of out-of-band agreements that ensure that both Federations can collaborate properly.

4.1. Abbreviated terms

- AS Authorization Server
- M2M Machine-to-Machine
- FM Federation Manager
- IDP Identity Provider
- LDAP Lightweight Directory Access Protocol
- OIDC OpenID Connect
- RO Resource Owner

- SAML Security Assertion Markup Language
- SCIM System for Cross-Domain Identity
- SP Service Provider
- UMA User Managed Access
- XACML eXtensible Access Control Markup Language

Chapter 5. Overview

This Engineering Report is comprised of the following main sections.

Section 6 analyzes and further explains the concepts of federation, governance and deployment models, and lays the groundwork for the subsequent sections in this ER.

Section 7 discusses the need for a Federation in terms of management of its resources (such as users, W*S services and data).

Sections 8 and 9 present the two Federation Managers explored and connected in this activity. These components, provided by Deimos and rasdaman respectively, are nicknamed FM #1 and #2 throughout the document. They comprise the main technical components with the purpose of enabling all the Federation needs discussed in section 6. Both components have been designed and implemented during this Testbed using current standards, federation needs, and security as their functionality drives.

Section 10 includes a summary of all the Testbed components involved in the federated clouds architecture described in previous sections and their federation-relevant connectivity and functional tests.

Chapter 6. An Introduction to Federation

As the geospatial community continues to find ways to efficiently share resources within collaborative environments, the requirements arising from this challenge are strongly tied to the concept of "Federation". While the concept of some alignment of data and/or service offerings seems to be commonly accepted there are wide variations on the facets and degrees of alignment given the varied nature of the requirements.

Therefore, in an attempt to narrow down the federation concepts to the level of accessibility and comparability, this ER emphasizes federation facets that are more common in the OGC standards ecosystem. There is an acknowledgement that there may be further relevant aspects that could be addressed in future work.

As a conscious choice, in this Testbed activity two federation managers were combined. Each manager follows different paradigms, thereby allowing a side by side comparison of effects:

- D143 Federation Manager #1 (FM #1) offers a freestanding service orchestrating authentication and authorization of a set of independent services that just have to be registered. The services can either be registered and controlled by their respective Resource Owners, or they can be managed by the central federation manager who is able to define authentication and authorization rules for all. This approach provides the means to delegate authentication to external providers, and treats each external Federation Manager as a selfstanding service with its representative Resource Owner.
- D142 Federation Manager #2 (FM #2) is a tightly integrated component of the rasdaman datacube engine and, as such, interoperates only within rasdaman federations (which, in this case, is a federation between several rasdaman instances in the US and in Europe). The federation manager is decentralized without a single point of failure (much like a DNS). Authentication can be managed through rasdaman or some external identity provider, in this case: FM #1. Authorization remains strictly local with each node, every data center retains full autonomy on access rules.

As part of the interoperability experiment, FMs #1 and #2 have been coupled in the sense that FM #2 accepts FM #1 as external identity provider, thereby combining FM #1 and #2 realms into one single user space. In addition to this, FM#1 provides a proxy service that allows FM#2 to interact further in terms of resource discoverability.

6.1. What is a federation?

While federation can mean different things based on different perspectives, some definitions have been collected which address different facets of federation:

- NIST [3]: In the simplest terms, federation means to enable interaction or collaboration of some sort. Of course, this can mean very different things in different use cases, in different application domains, and at different levels in the system stack.
- [Wikipedia definition of Federation in IT](https://en.wikipedia.org/wiki/Federation_(information_technology)) [https://en.wikipedia.org/wiki/Federation_(information_technology)]: a group of computing or network providers agreeing upon standards of operation in a collective fashion

- [Wikipedia definition of Federation in Databases](https://en.wikipedia.org/wiki/Federated_database_system) [https://en.wikipedia.org/wiki/Federated_database_system] lists these key properties: transparently maps multiple autonomous database systems into a single federated database; constituent database systems remain autonomous; no actual data integration (maybe different schemas, even different QLs underneath, but hidden from the user)

Notably, these definitions in increasing conciseness go from "interaction or collaboration of some sort" to "agreement on standards of operation" and finally concrete service properties.

During Testbed-15, the following (non-exhaustive) list of federation facets emerged:

- Identity Federation. Allows sharing (on a secured manner) identity credentials with Service Providers (or other Identity Providers) that can either be internal or external to a specific Administrative Domain. FM #1 offers global identity management, whereas FM #2 offers local identity Federation plus tapping into some external identity manager. However FM#2 does not offer identity management to non-members.
- Authorization Federation. Allows common access control. For example, determine for some given user the rights on access, processing, and so forth they have on the resources available in the federation. This authorization may be managed centrally (FM #1) or on-site (FM #2).
- Resource Access Federation. Provides discovery, registration and access mechanisms for resources. Access to a particular resource may require users to address the proper source hosting it (location opaque federation - FM #1) or, alternatively, accessing any host in the federation may take care of providing the resource on request (location transparent federation - FM #2). Resources can vary in granularity. A resource could be data behind an OGC Web Service, an OGC Web Service operation or an OGC Web Service as a whole.
- Service Federation. Provides access (as a special, trivial case), filtering, processing, or general analytics on resources in a location transparent manner. In particular, distributed data fusion is accomplished transparently. Users send a request which may involve data from different hosts, and the federation orchestrates dynamically and returns only the final result to the user. Location-transparent federation is offered by FM #2, mainly due to the fact that it can be directly coupled with W*S Services (deliverable D147).

Instead of providing its own definition of Federation, this document puts forward the different approaches to Federation, commonly named Federation Facets. These Facets have in part also been identified in previous OGC Innovation Program Testbed activities. As this ER is heavily focused on the Security aspects of Federation, some of these layers may be detailed in a greater extent than others.

6.2. Security and Federation

Please note that the concepts of Federation and Security do not necessarily need to coexist within specific environments, since this will depend on which approach is implemented in a Federation Layer that is being utilized by the Administrative Environment that deploys the Federation components. Nevertheless, any Federation Architecture should have its own embedded Security Architecture.

Some of these approaches for implementing Federation Layers are directly tied to specific Security

topics:

- **Identity Federation.** This specific kind of federation is directly recognized as a knowledge domain and critical aspect of Security Architecture, since it recognizes Authentication and Data Protection needs.
- **Resource Federation.** In this case, any resource that belongs to a Federation should be part of a pre-defined Access Control and Access Management architecture. These are also knowledge domains within the concept of Security Architecture, providing Authorization and Accounting within a Federation.

Although other Layers are mentioned in this section, the focus of this ER is on the Identity and Resource Management and the security use cases that they enable.

6.2.1. Identity Federation - Authentication

One of the key features of a collaborative environment is the ability to easily identify the participants, enabling an Authentication process behind any actions performed within the Federated environment. To provide this federation layer, all participants need to agree on a common set of tools and standards that will provide the necessary interoperability to share user information securely.

The main concepts that this Federation layer brings to the table are:

- **Identity Providers (IDP).** These are components within the Federation that store and release end-user information based on end-user and Service Provider needs. Whenever an unknown end-user tries to authenticate within an Identity Provider, it is possible to request information from other Identity Providers within the Federation that may have information about the specific end-user. When classifying Identity Providers using more formal communication architectures, they tend to act as servers that provide an authentication service.
- **Service Provider (SP).** These components mainly retrieve information about the end-user to provide a service to the Federation. This release of information is usually constrained by end-user directives, privacy policies and terms of use. When classifying Service Providers within more formal communication architectures, they tend to act as clients to the Identity Providers, but also as servers to the end-users providing their specific service.

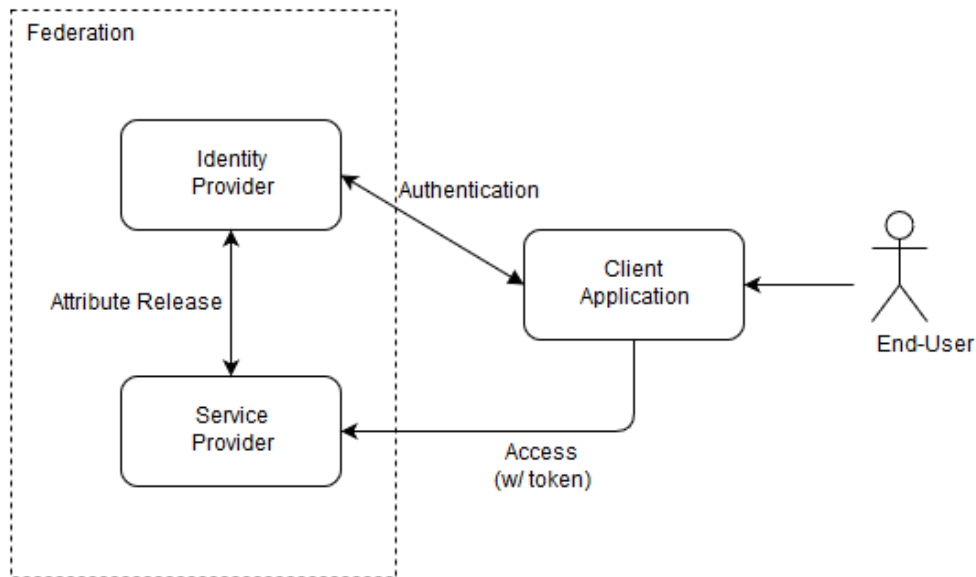


Figure 1. Identity Federation Architecture

This layer allows users to share identity credentials with Service Providers (or other Identity Providers) which can either be internal or external to the Federation. This exchange of information needs to be heavily secured, given the nature of the data being passed between entities.

6.2.2. Resource Federation - Authorization

Another key feature of a Federation is the ability to provide the means to discover, register and access resources. These capabilities need to be consumed not only by end-users but also internally in Machine-to-Machine interactions. Although standardization of the means to discover, register and access resources is not a clear necessity from the end-user's perspective, it is necessary at least for internal components to be able to propagate metadata about resources between themselves. This, in turn, facilitates the implementation of different governance and deployment models based on the Resource Access requirements that are put forward by all members of a Federated Environment.

Resources can be defined at any level within the service stack, allowing different types of federations. This ER focuses specifically on the aspects of Service Federations, and in some aspects also Platform Federations, since both their applicability domains correlate well with OGC Web Services [1].

6.2.2.1. Infrastructure Federations

Federation instances can be found at the Infrastructure level, where several private clouds offering Infrastructure-as-a-Service (IaaS) agree to allow virtualization of their resources on a transparent manner. This allows their user communities to make use of their resources with a greater deal of flexibility [2].

6.2.2.2. Platform Federations

Federation instances that run at the Platform level mainly aim at providing a set of services that utilize several platform aspects in a transparent manner, consisting of several platform providers working in unison.

Leaving the realm of security and identity, an example is to federate other kinds of platforms such as databases. The result of these federation activities is the possibility for transparently mapping several autonomous databases into a single one. This approach allows decentralization of resources and provides the means to perform single queries over the federated system, applying abstraction methods that in turn allow heterogeneous systems to work in unison.

6.2.2.3. Service Federations

The main concepts that this Federation layer provides are:

- **Resource.** This concept serves as an abstract representation of any added value item within a Federation. A resource can vary in granularity depending on how its Resource Owner wants it to be perceived by Federation Members. As an example of this granularity, a Web Processing Service (WPS) could be interpreted as a resource in the following levels:
 - Service Level. The whole instance is considered as a resource, meaning that rules for access control affect the whole service. The service can be registered and discovered as a base URI.
 - Operation Level. Specific operations within the service (i.e GetCapabilities, DescribeProcess, Execute) are considered resources, meaning that some may have different rules for access control than others. The operations are registered and discovered as endpoint URIs.
 - Example: For a WPS instance, the Execute operation is protected with an access policy, while the GetCapabilities and DescribeProcess could remain open
 - Offering Level. Specific items within service operations are considered as resources. The operations are registered as URIs with more specific content.
 - Example: For a WPS instance, a Corner Detection algorithm can be protected with a set of policies completely different from other processes. A CSW service could implement this at the collection level, a WFS service at the Feature level, etc.
- **Resource Owner (RO).** This entity represents a special case of End-User that retains governance over a set of resources (Service Provider), registering them, and deciding how they are discovered and accessed. The role of Resource Owner varies greatly depending on the Governance Model that the Federation uses.
- **Authorization Server (AS).** This component can appear next to an Identity Provider or be bundled together with a Resource (Service Provider) to implement Policy Enforcement and Decision Points and to provide means to discover and register resources. Authorization Servers not only provide a service to Resource Owners and End-Users since they also need to perform M2M operations to keep the catalog of resources consistent within the Federation. This consistency not only affects the discoverability but also to the access policy enforcement.
- **Access Policy.** These are sets of rules for resource access, usually defined as the Policy Decision Point (PDP). Resource Owners pair them together with resources to provide Resource Access Management.

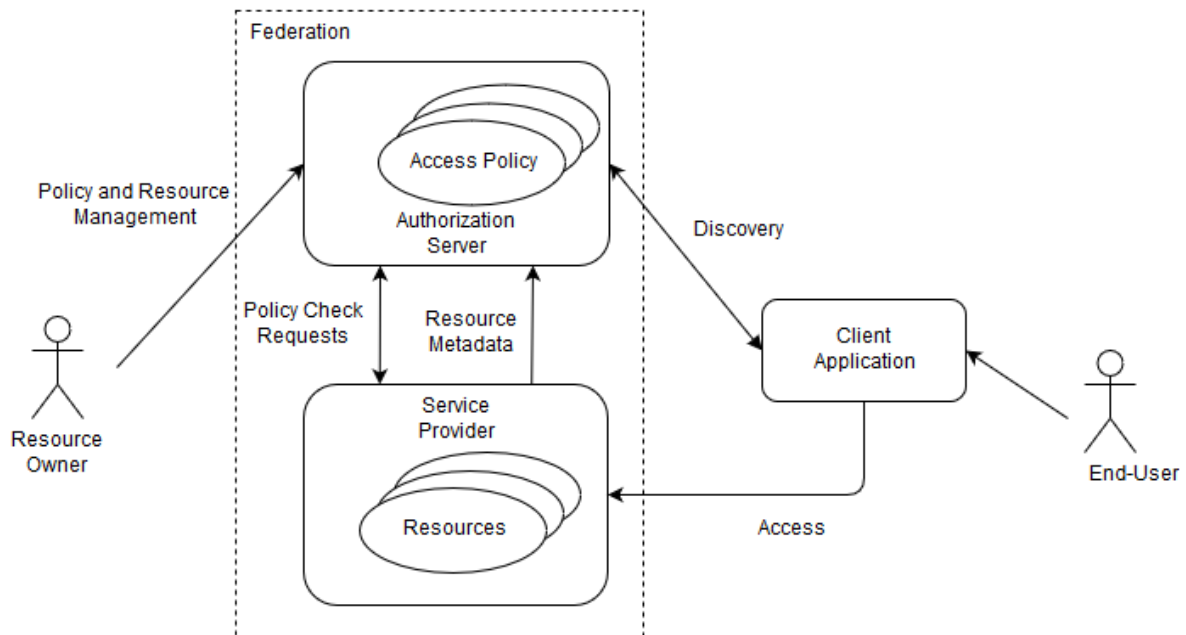


Figure 2. Resource Federation Architecture

The core functionality of this layer provides Resource Owners with the means to publish their resources securely. This allows the whole federation to discover and access those resources, while also retaining control over them. The security implications of these operations require a level of interoperability that can only be achieved by adopting state-of-the-art standards such as OAuth2.0, OpenID Connect or User-Managed Access.

6.2.3. Data Federations

Many of the Service-based Federations can be viewed as intermediate layers with the ultimate goal of providing Data Federation. In that sense, the "data" is the "resource". Federations may have different types of data to be shared with their independent federation agreements:

- Membership information (user profile information)
- Usage/Accounting information
- Inputs and Outputs used by Services within the Federation

All of these data types, and others that could be specific to a federation, can be subject to Federation Policies, Data Regulation Laws, and involve all stakeholders of a Federation

6.2.4. Service Federations

While in a Data Federation a service may be offered, and some processing will be involved when answering a request. In a Service Federation processing plays a central role. Typically, this processing is more involved and requires significant resources on the server side.

The advantage for the user is that the "problem solving" task is shifted to the server which just returns the overall result. Distinguishing criteria are the degrees of freedom of service tasking, ranging from invoking fixed functionality to flexible query languages.

A particular use case is data fusion between federation nodes where a user requests a combination

of data sets residing on different federation nodes. Several degrees of location transparency can be distinguished, such as:

- The user sends a processing request to an identified federation member which answers.
- The user sends a processing request to the federation (i.e., any node in the federation), and federation members forward the request to a suitable member node (such as a node holding the data needed) which answers.
- The user sends a processing request to the federation which requires distributed data fusion, and federation members orchestrate processing in an ad-hoc fashion to generate the answer.

6.3. Deployment Models and Governance

The concept of federation entails much more than the transparency and sharing of resources that happens at the end-user level. In most cases, the creation of a Federation requires several high-level entities to agree on how these resources are shared, who is their owner, and which are the basic rules and requisites to be part of the federation.

A governance model represents all these agreements (which in themselves are the governance framework) and policies. Therefore, a governance model allows clearly defining all entities, practices, and procedures within the federation. These governance models, when put in practice, result in several deployment models that reflect the degree of openness, the possible stakeholders and their capability to influence the federation.

6.3.1. Resource Owners

The role of Resource Owners within a federation can vary between different governance models, depending on the amount of control they retain over the resources they are providing to the federation itself, and whether or not they retain rights over them (mostly for billing or confidentiality purposes). A possible classification of Resource owners could split them into two categories:

- Active Resource Owners retain control over the management access rights to the resources and are responsible for managing them as long as they remain within the basic rules and requisites that define the Federation.
- Passive Resource Owners forfeit control over the management of access rights to resources, either by delegating that role in a centralized figure (administrator) or by making all resources free and open (if the federation governance model allows this to take place).

6.3.2. Onboarding Requirements

Federations and applying Service Providers need to have their interests aligned to be able to successfully collaborate and interact with each other.

In some cases, the Resource Owner is willing to provide to the Federation a set of resources that may have confidentiality restrictions. The Resource Owner may also need to retain accountability and billing over the usage of their resources, even while the resource remains a part of the Federation.

The federation governance model needs to allow for this situation to happen. The Federation can either reject their participation due to its inherent constraints (an example would be a federation with the purpose to provide only free and open resources) or it can provide the Resource Owner with the tools to enforce the confidentiality rules (acting as an Active Resource Owner).

On the opposite side of the spectrum, open resources can also be rejected by a Federation if one of its pre-requisites is to avoid competition deemed as unfair between their members.

As with these last two examples, many others could arise during an onboarding process and, while they require human interaction to be solved, it is necessary to provide the technical means to facilitate onboarding processes as much as possible.

6.3.3. Federation Maintenance Matureness

Most of the existing Federated environments start with an out-of-band agreement between stakeholders and, given this agreement, several internal resources are generated to support well-defined procedures. These resources can be split into several categories:

- **Documentation** including outreach material and advancing a common understanding of what the federation is all about.
- **Interoperability components** that aim at facilitating the on-boarding process and its associated technical cost by using state of the art standards and architectural solutions.
- **Automation of procedures and tasks**, reducing the amount of required interaction between Resource Owners

It is also important to take into account that User Education and Awareness is one of the main Security knowledge domains that a Matured Federation should aim for, mainly through the aforementioned resources.

Chapter 7. Federation Management Architecture

7.1. Introduction

During the Testbed-15 activity, an effort was made to demonstrate the complexity and issues tied to the concept of Federation. This was not only between Services/Resources and the Federation but also the interaction between Federated environments (also known as Federation of Federations or inter-federation). To do so, the concept of Federation Manager is put forward and explored in detail in the following sections.

The main objective was to simulate a scenario whereby two Federated Environments interact with each other employing their Federation Manager components and expose their endpoints to provide membership, resource access, and policy management capabilities. These capabilities, from the point of view of the Security domain, provide the necessary layers of Authentication and Authorization. In addition to this, each Federation Manager internally handled one or more services. Both Federation Managers represent different levels of maturity in different areas. This adds to the complexity of making them interact with each other.

The NIST Cloud Federation Reference Architecture [3] puts forward the following definition of Federation Manager:

The Federation Manager is the conceptual entity that manages one or more Federation Instances, or simply federations, among two or more Administrative Domains [...]. It could provide several brokerage-like functions, but establishing and operating a federation across multiple sites requires that it perform several critical management functions over the lifespan of a federation instance. This is far beyond the traditional brokering function of just connecting a buyer and seller, after which the broker's function is done.

In general terms, a Federation Manager should aim to produce the most matured Federation possible. Maturity of a federation, in this case, is measured by how much of the governance work is automated (leaving only the remaining stakeholder human interactions required to solve governance issues) and the degree of Security achieved in their interactions when providing the Access Control and Identity capabilities.

Taking this definition into account, several functionality aspects are required from a Federation Manager. To enable inter-federation, these aspects need to be served not only to users but also to other Federation Managers.

7.2. Deployment models

The NIST Cloud Federated Clouds Reference Architecture [3] identifies several Federation Manager (FM) deployments, that assume a series of properties mainly based on their location within their respective Federation environments, and their interactions with other Federation Managers.

7.2.1. Topology

The Reference Architecture identifies three main deployments in this aspect:

- Centralized FM Deployments only implement interactions with their sites and their Service Providers. Such deployments are not expected to interact with other Federation Managers. In this respect, they do not tackle the issue of inter-federation, which is one of the main subjects of the Engineering Report.
- Pair-wise FM Deployments are prepared to not only interact with Service Providers but also with other instances of Federation Managers. In that sense, these deployments allow to establish one-to-one connections with each other.
- Complex FM Deployments, are basically more complex topologies that are based on combinations of the other two deployment models mentioned above.

7.2.2. Relationship with Sites

The NIST Reference Architecture identifies two main possibilities that mainly affect Governance Rules:

- External: The FM acts as a standalone component within the Federation.
- Internal: The FM is part of a specific site.

In the first possibility, a third party is required in order to operate and manage the Federation Manager. If Resource Owners are passive, all sites need to agree to trust in the third-party owning the Federation Manager, while in the case of active Resource Owners, the trust relationship remains but only in the ability of the Federation Manager owner to provide the Federation Management services properly.

The second possibility results in less complex out of band relationships, and forces Resource Owners to keep an active administrative stance.

A third case, distributed FM within a federation, is not yet addressed by the Reference Architecture. Here participating nodes retain full autonomy, but may interoperate where deemed feasible, without extra overhead for the service maintainers.

As before, mixed situations are possible and were explored in this Testbed.

7.2.3. Federation Manager Hierarchy

The NIST Reference Architecture also stipulates that there might be hierarchies between Federation Managers:

- Parent/Child relationships in the case that governance of one Federation Manager is define by the other.
- P2P relationships: where there is a trust relationship that enables coordination.

7.2.4. Deliverable deployments

During this Testbed, two Federation Management deployments were delivered:

- Federation Manager #1. A standalone component within the Federation.
 - Pair-wise ready deployment that could interact efficiently with other Federation Managers, by providing a Federation Management API.
 - Configurable to be a Third-Party deployment or several instances could be deployed for any number of sites.
 - Based on a push architecture with access scopes that allows its usage on different hierarchy structures.
- Federation Manager #2. A distributed manager with automatically synchronizing instances running on each Service Provider.
 - Pair-wise ready deployment only for compliant Service Providers. No global Authorization Management as member nodes retain full autonomy on their data. Membership and Resource Access functionality accessible by external (non-compliant) Federation Managers.
 - Authentication possible via external or built-in Identity Manager.
 - Tight, transparent integration with instances running within Service Providers
 - P2P relationships only between Federation Managers running on compliant Service Providers.
 - Full location transparency for clients.

7.3. Federation Manager to Service Provider Interactions

From the point of view of Service Providers, any Federation Manager is expected to provide the means to efficiently interact with End-Users. This means providing Authentication and Authorization capabilities as well as interaction with the Resource Owners who contribute with their services in the Federation.

Ideally, a Federation Manager should be as invisible as possible to clients while enabling the use of dedicated protocols and APIs for the Service Providers (and clients) to fulfill the use cases listed in the following sections, for all stakeholders. Bear in mind that the role of Resource Owner could represent different stakeholders depending on the Governance Model as stated in [Governance](#) is important.

7.3.1. Federation Membership Management

7.3.1.1. Use Cases

- End-Users authenticate themselves using several authentication mechanisms.
- Resource Owners request Membership information about specific users.

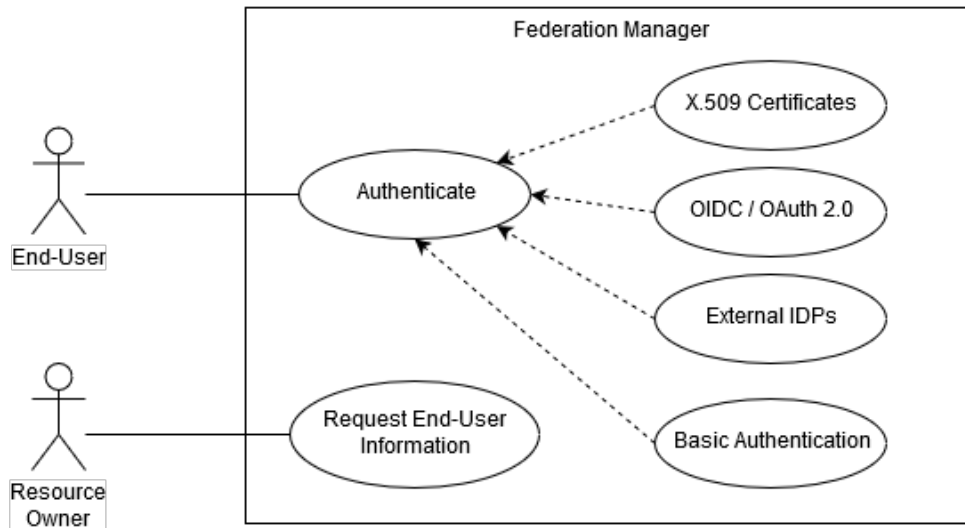


Figure 3. Membership Management Use Cases Summary

7.3.1.2. Maturity ad-hoc requirements

- End-Users need to be able to share their membership data within a Federation that complies with all Data Regulation policies that may apply.
- Resource Owners within a Federation should not be able to freely access membership information without user consent.
- Authentication mechanisms should allow as much interoperability as possible.
- All outgoing membership information (from the Federation to external entities) should be traced and require end-user consent.

7.3.2. Federation Resource Management

7.3.2.1. Use Cases

- End-Users (and Client implementations) discover and access resources.
- Resource Owners register public and restricted resource references.
- Resource Owners grant and revoke access to the Resources they own (modifying membership dedicated attributes).
- Resource Owners modify resource access restrictions by applying different access scopes (and other complex policies).

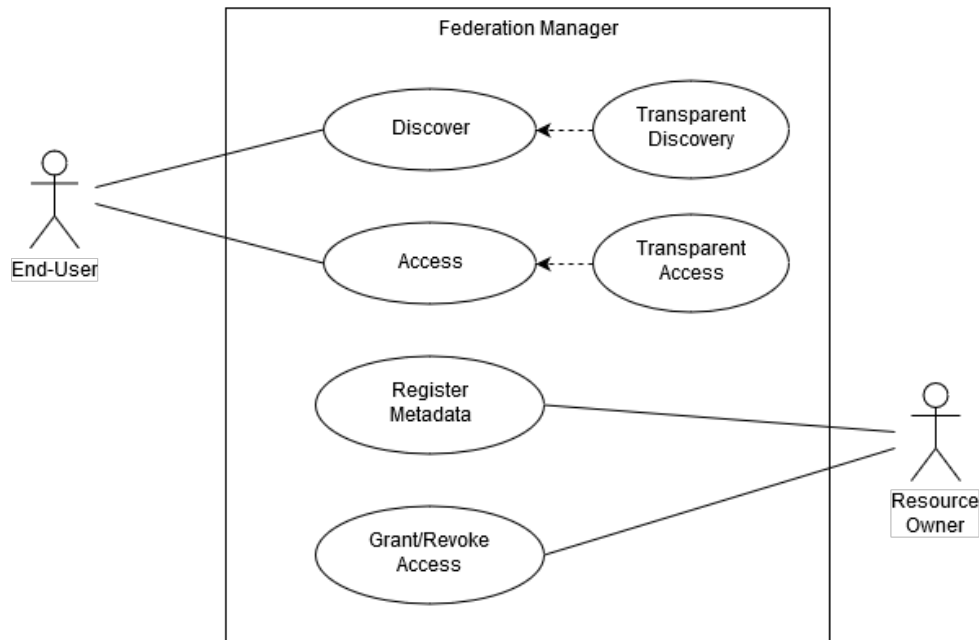


Figure 4. Resource Management Use Cases Summary

7.3.2.2. Maturity ad-hoc requirements

- Resource Owners should be given the necessary tools to be able to grant and revoke access to users (i.e. through administration portals and corresponding API endpoints).
- For Resource Owners to avoid unwanted metadata propagation, discoverability of resources should also be protected.
- A Federation Manager could provide an abstraction layer presenting a single, seamless data and processing layer. For example, location transparency (one contact point for Client Implementations that orchestrates incoming requests across the nodes of a single Federation Service).
- A Federation Manager could provide an abstraction layer that enables transparent resource access (one contact point for Client Implementations that utilizes several Federation Services).

7.3.3. Federation Policy Management

7.3.3.1. Use cases

- Resource Owners define access policies
- Resource Owners attach policies to resources

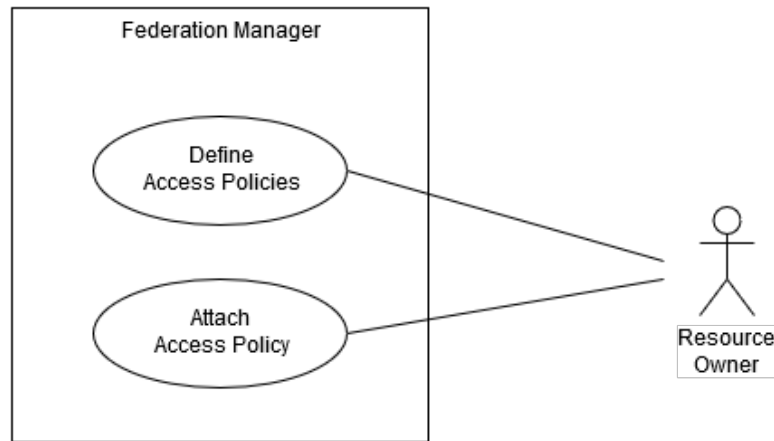


Figure 5. Policy Management Use Cases Summary

7.3.3.2. Maturity ad-hoc requirements

- Policies can implement granular access based on a variety of restrictions (not only attribute-based).
- Resource Owners should be given a flexible and customizable policy definition language (XACML or similar).

7.4. Federation Manager to Federation Manager Interactions

In addition to the previously mentioned use cases, a Federation needs to efficiently communicate with other Federations, thus enabling inter-federation. Both membership attributes and resource metadata need to be shared and modified in a controlled manner to achieve convergence between any number of Federation Managers.

7.4.1. Federation Portability and Interoperability

7.4.1.1. Research

Synchronization in a distributed system has been studied intensively in Computer Science (see, [4] and [Distributed Database Systems](https://en.wikipedia.org/wiki/Distributed_database) [https://en.wikipedia.org/wiki/Distributed_database]), and concepts and tools are readily available in the realm of Distributed Computing.

7.4.1.2. Standards

An obvious approach to portability and interoperability is to utilize state-of-art standards that enable the previously mentioned use cases:

- OAuth/OpenID Connect.
- User-Managed Access.
- System for Cross-Domain Identity.
- OGC Standards Baseline.

The use of the first three mentioned standards is especially useful when a Federation Manager treats other Federation Managers as if they were Service Providers with elevated privileges. There is also the option for Federation Managers to see each other as authoritative (trusted) players. This would however require new protocols.

In any case, the use of these standards does not solve Federation Management issues, since most of them specify too many endpoints and complex interactions. During this Testbed activity, the participants identified the need to provide an abstraction layer on top of these standards in order to tackle the issue of inter-federation.

As with any other Machine-to-Machine (M2M) interactions, all possible inter-federation architectures can be reduced to a combination of "push" and "pull" operations and defined using the set of conditions that are required for two components to be able to interact. This is also known as "neighboring conditions" in the realm of networking.

The necessary conditions for two Federation Managers to interact (that is, to be neighbors) are:

- An out of band agreement between the representatives or owners of both Federation Managers. This agreement needs to specify which information is shared under which specific access constraints (i.e. scopes) and should make sure that both parties agree on a common overarching goal for the inter-federation. This is similar to the onboarding of new members in a federation.
- Clearly defined interaction protocols (such as APIs) that ideally utilize well-known standards
- Allowed cross access, meaning that each Federation Manager can interact with the endpoints of the opposed Federation Manager in a trusted manner.

7.4.1.3. Push architectures

In "Push" architectures, a source component chooses when to send its data to neighboring target components, following their endpoint specification to do so.

- Benefits
 - The owner of the data retains control on how much information is shared, which are the targets of the communication and when this process takes place.
 - The information stored in each of the Federation Managers is reliable at almost all times. The only case when this information does not converge is whenever a Federation Manager decides not to push membership or resource information to neighboring Federation Managers.
- Drawbacks
 - A "push" architecture requires a series of interoperability ad-hoc components that adjust to the needs of Federation Managers that receive membership and resource data, but also make it possible for others to inject their data if done insecurely. This point can be resolved by either standardization efforts that provide specifications to the specific endpoints and interactions (such as those listed on Standards subsection above) or by providing extensive tooling for Federation Managers, such as a plugin or proxy for each Federation Manager implementation type.

7.4.1.4. Pull architectures

In "Pull" architectures, the source component is passive and waits for other components to request information.

- Benefits
 - From the point of view of each Federation Manager, this architecture comes with all the benefits of an eager/lazy interaction, that requires less development effort and facilitates integration and interoperability.
 - The owner of the data, as with Push, retains control over what is shared
 - An acting component (i.e., addressed by a client request) can decide whether to preemptively pull information from a source component, or pull on request, or pursue some other policy (like a mix of the previous ones).
- Drawbacks
 - The owner of the data loses control over **when** the process for releasing information takes place.
 - A pull architecture presents scalability and convergence issues. Usually, a polling period for the requests is put in place to keep an updated status of the other Federation Managers. This process causes data divergence in between polling operations that can only be solved increasing the frequency of requests, and eventually running into scalability issues.

Chapter 8. Federation Manager #1

8.1. Design Overview

The amount of expected functionality from a Federation Manager increases the complexity of previously tested deployments (such as the one utilized in Testbed 14 [5]). The structure for Federation Manager #1 is shown in the following figure:

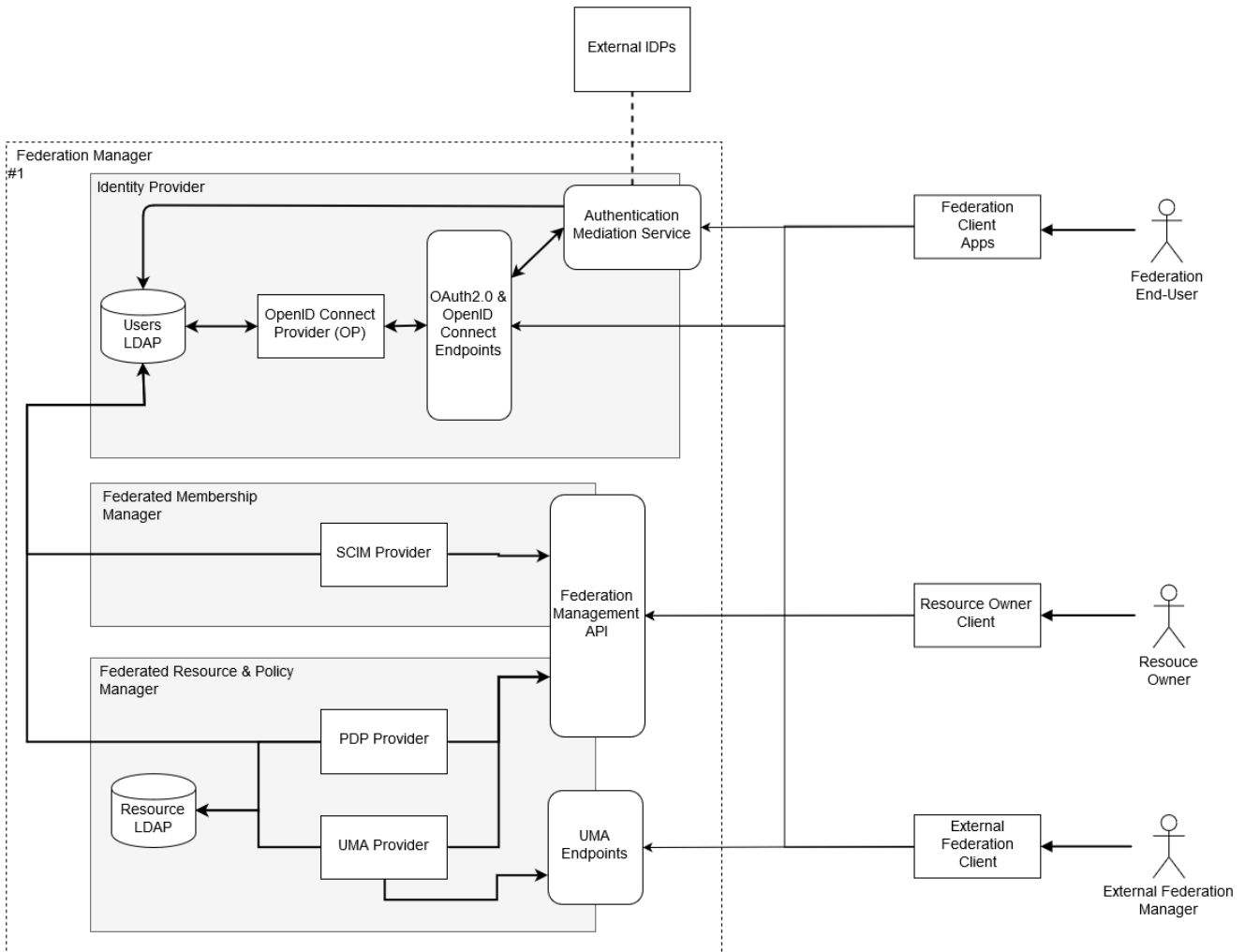


Figure 6. Federation Manager #1 Architecture

Federation Manager #1 utilizes state-of-the-art standards to provide Authentication and Authorization capabilities using:

- An Identity Provider implementation compliant with OpenID Connect (OIDC), handling **Federated Membership Management**, that can be used by both Services and Clients
- An Authorization Server implementation based on User-Managed Access (UMA) handling **Federated Resource and Policy** management tools for Resource Owners, and access management for Clients
- Back-end services as means of storage of membership and resource information (LDAP based in this specific case)
- A Federation Management API specification providing an abstraction layer that can be used by

A break-down of all components and the use cases enabled by each of them is provided in the following sections.

On the other hand, three main types of Client Applications are identified based on their interactions with the Federation Manager endpoints and the End-Users they support.

- **Client Applications for End-Users.** These clients reside within the federation and communicates with the Identity Provider to provide authentication, UMA Endpoints to participate in the resource access flows, and the Federation Management API to discover resources.
- **Resource Owner Clients.** These clients also reside within the federation and communicate with the Identity Provider to authenticate Resource Owners. After authentication, Resource Owners are able to exercise their use cases against the Federation Management API.
- **External Federation Clients.** These clients are essentially the same as Resource Owner clients. The two key differences are that they reside outside of the federation and are accessed by representatives of a specific external Federation for the purposes of inter-federation.

8.2. Functionality

This section focuses on the use cases mentioned in [Federation](#) and how the Federation Manager aims at fulfilling them.

8.2.1. Federation Authentication: Membership Management

Federation Manager #1 acts as an OpenID Connect provider, a figure within a federation where user credentials and attributes are stored. The specifics for this functionality were exercised in Testbed-14. Both the Security ER [5] and the Federated Clouds ER [6] provides extensive explanations on how information flows from clients to services through the Identity Provider.

In addition to this functionality, Federation Manager #1 also provides Mediation and Delegation Capabilities, thus allowing the secured release of attributes within scopes of access, either internally or externally to other Federations. This can be achieved by leveraging OIDC standard API endpoints, described the Annex of the Testbed-14 Security ER and listed under the discovery document (.well-known/openid-configuration).

Going even further Federation Manager #1 provides certificate authentication using TLS with X.509 Certificates. This type of authentication allows other Federations to sign two main types of certificates:

- **Personal PKI x.509 certificates,** released to end-users. These could be added to a web-client (such as browsers) and used in order to authenticate access into the system.
- **Intermediate PKI x.509 certificates,** released to external Identity Provider. These could allow them to extend the chain of trust to the users within their federation, so that they can utilize them on the Federation Manager.

The following use cases have also been exercised by the Federation Manager #1:

- Users: Authentication using internal or external credentials (inherited from Testbed-14) and also with X.509 certificates.
- Resource Owners: Request membership information about specific users for its use in Services

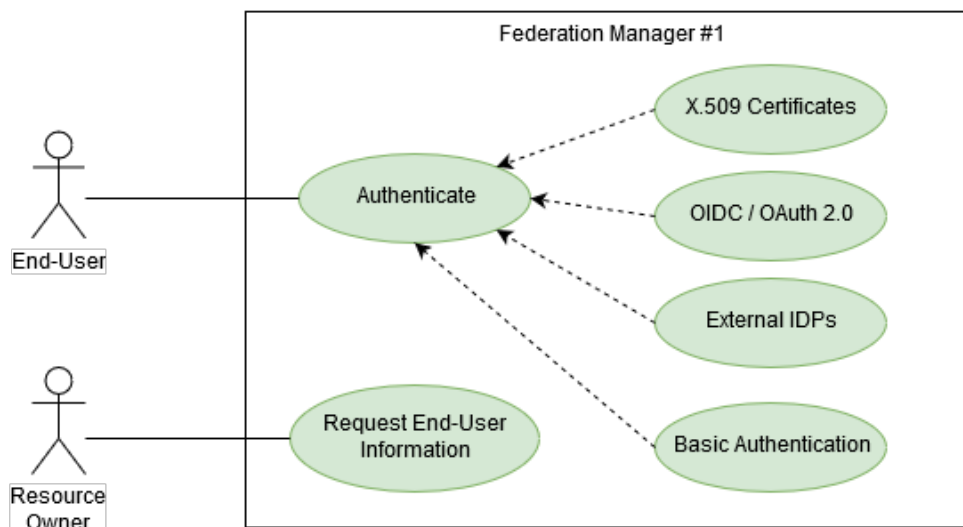


Figure 7. Membership Management provided by Federation Manager #1

8.2.2. Federation Authorization: Resource and Policy Management

Federation Manager #1 extends Membership Management by providing Resource Management and Policy-based Access to all Resource Owners. Resource Owners are granted privileges based on their usage of the Federated Membership Management tools mentioned in the previous subsection, thus allowing them to register and protect resources using the Federation Management API component. The following functionality is provided by the Federation Manager #1:

- For Users/Clients
 - **Discover a resource.** Both resource owners and end-users are able to list the resources registered in the Authorization Server using the corresponding FM API endpoint (with their respective levels of visibility).

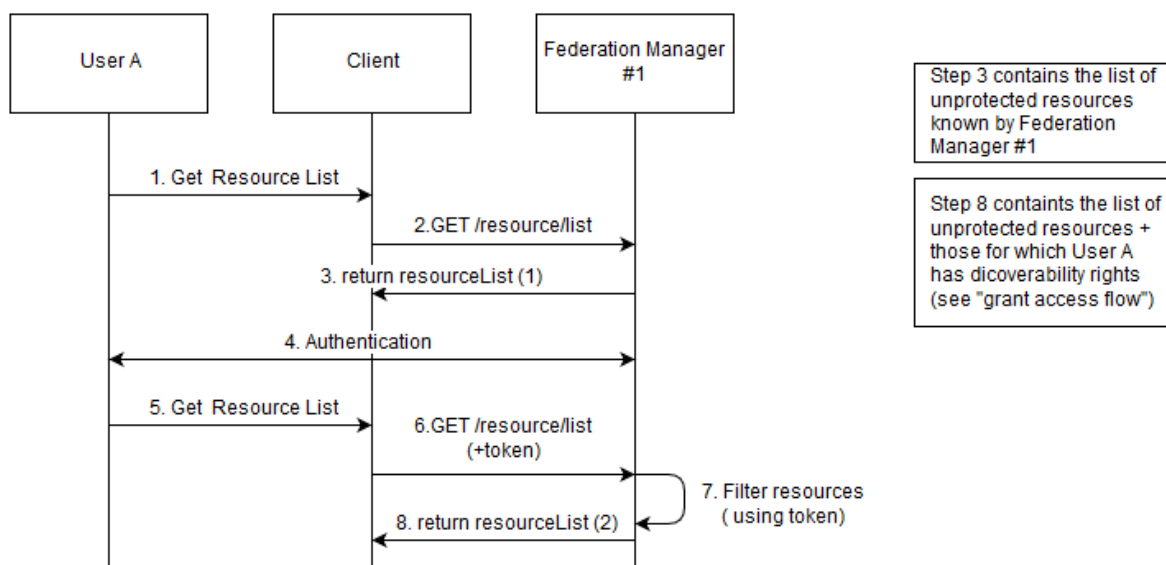


Figure 8. Federation Manager #1 Resource Discovery

- **Access a resource.** After protection, the web service that contains the resource can choose to reject access to the service and redirect the user to the Authorization Server with an access ticket. This ticket, added to user (or application) credentials, will be exchanged for a token.

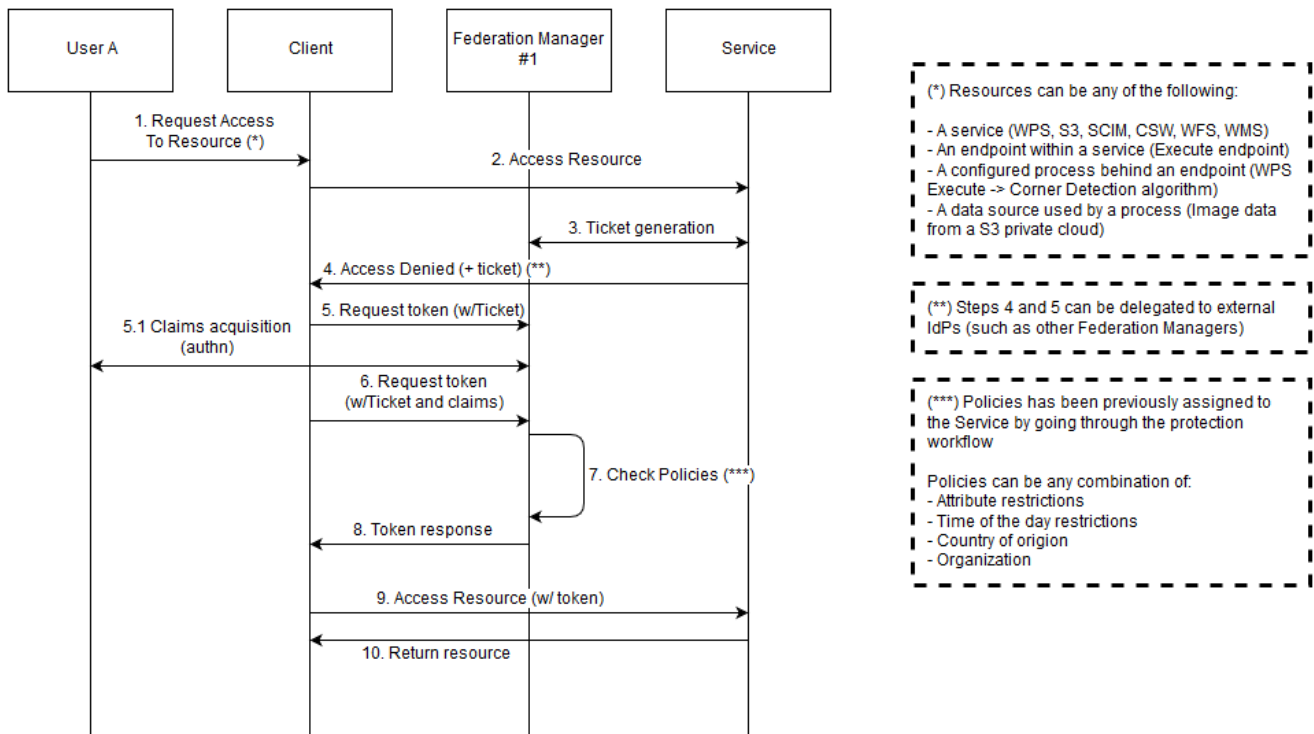


Figure 9. Federation Manager #1 Resource Access

- For Resource Owners
 - **Publish a resource reference.** A resource owner with Resource Owner credentials generated by the OpenID Connect Server will be able to publish their resources on an internal list within the Authorization Server using the FM API. This interaction can be achieved by systematic consumption of the FM API by a client internal to the Service, or by on-demand requests performed by the Resource Owner itself.

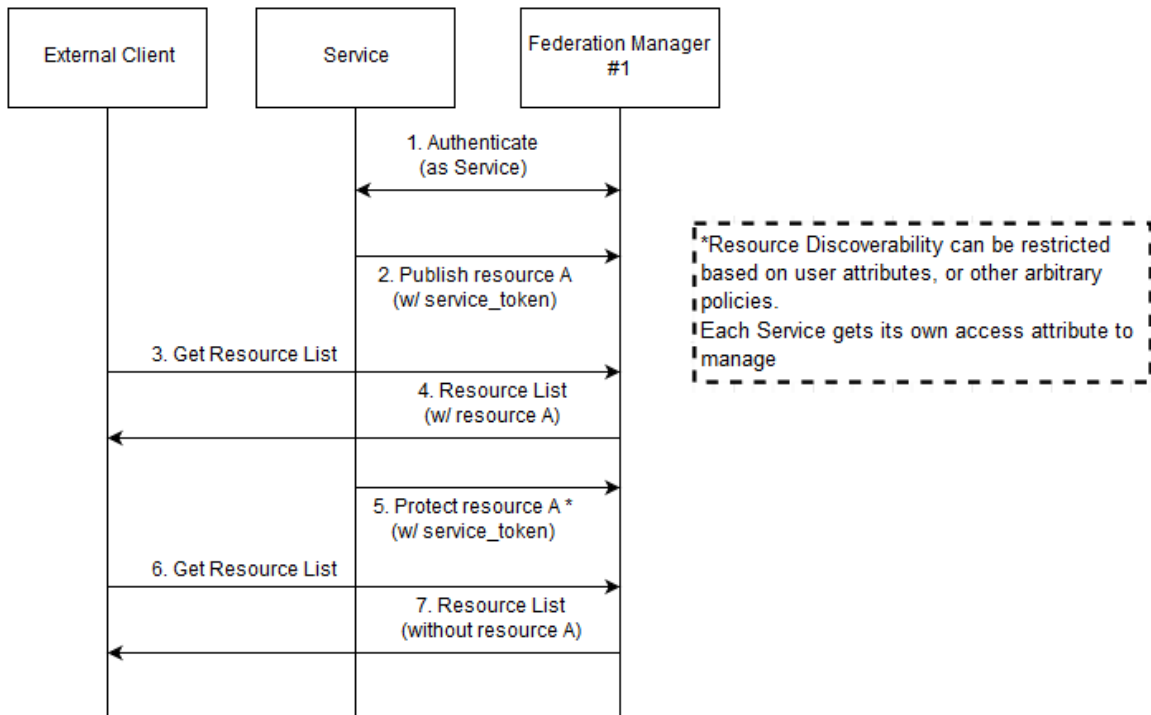


Figure 10. Publishing of resources within Federation Manager #1

- **Protect a resource reference.** Resource Owners can assign access scopes and predefined policies to resources, therefore restricting access. This step can be executed concurrently on the same request used to publish the resource.

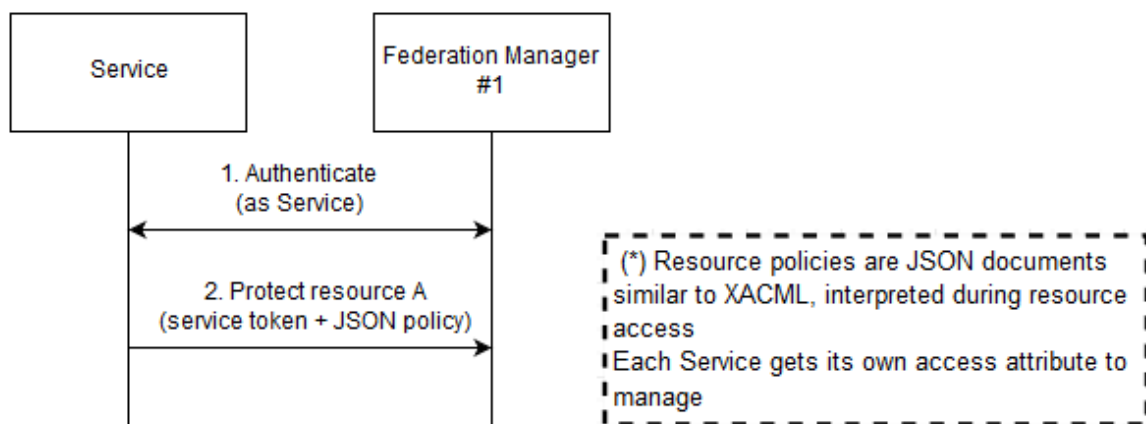


Figure 11. Federation Manager #1 Resource Protection

- **Grant (or revoke) resource access to users.** A resource owner (or an operator, depending on the governance rules set in place) can request access for published resources.

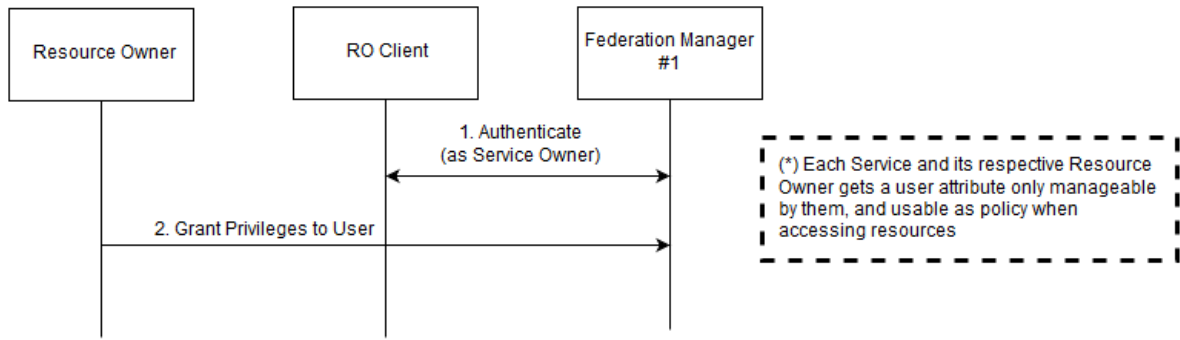


Figure 12. Federation Manager #1 Resource Access Grant

As a summary, the following use cases are covered by Federation Manager #1 (shown in green)

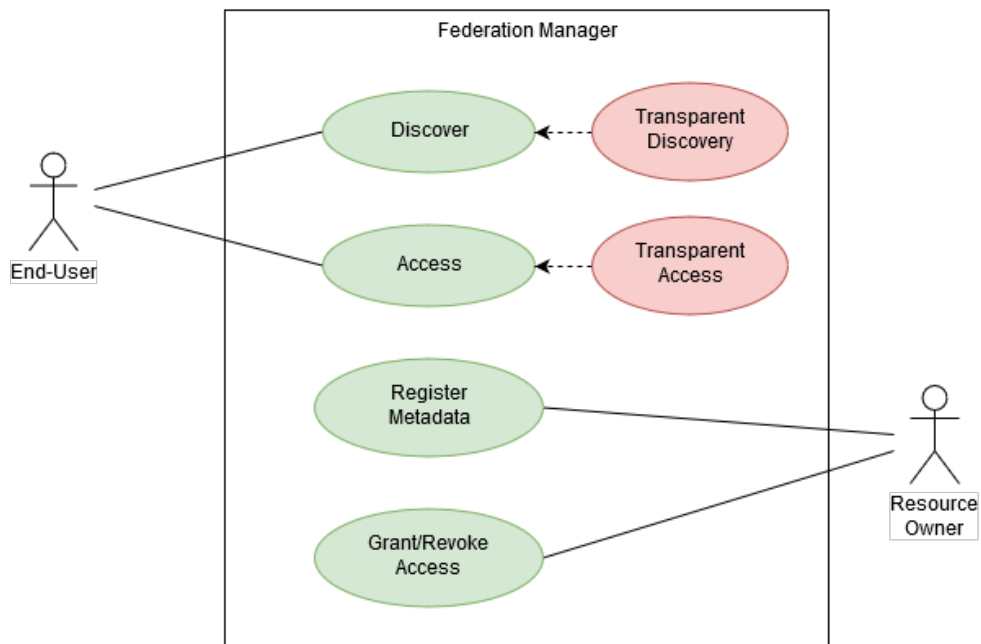


Figure 13. Resource Management provided by Federation Manager #1

8.2.3. Inter-Federation

This type of implementation for a Federation Manager requires other Federation Managers to interact within a "push" architecture. While the propagation of Membership Information is completely covered by already existing standards such as OpenID Connect, SAML or OAuth2.0, discoverability and access to resources across Federation Managers requires specific operations which are not provided by the User-Managed Access standard.

- **Publish a resource reference.** A local Federation Manager can have a list of "neighboring" Federation Managers that will receive an update to their list of resources whenever a new entry is created in the local Federation Manager. This propagation of registration and removal of resources results on "neighboring" Federation Managers acting as if the local Federation Manager is just another Service Provider. In this case, the Resource Owner would be the Federation as a whole (or in more private deployments, a Federation Manager administrator).

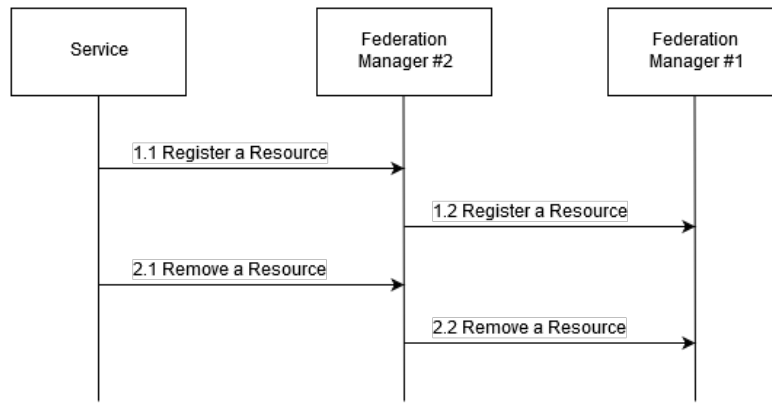


Figure 14. Resource Publishing across Federation Managers

- **Discover and Access resources.** The propagation of resource references across domains, requires Federation Managers to allow for the possibility of restricting discoverability based on specific discovery scopes. After discovery has been achieved, access to the resources can be exercised with the same procedure previously mentioned in [Federation Authorization: Resource and Policy Management](#).

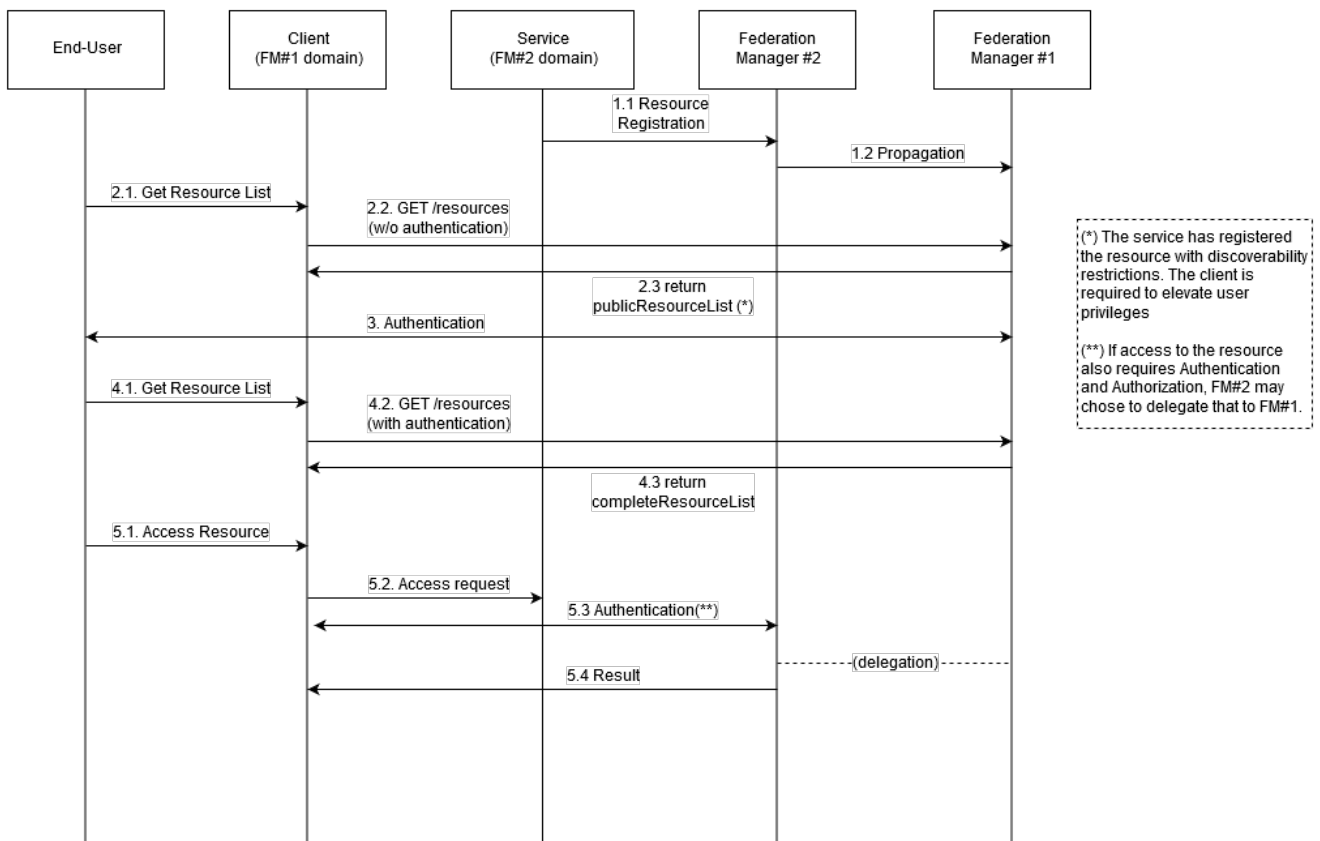


Figure 15. Resource Discovery across Federation Managers

These two specific inter-federation capabilities, both of which greatly benefit from the ability to **delegate End-Users authentication** to the Federation Manager on their home federation, eventually lead to the ability to create workflows that utilize services residing in different domains, with different access restrictions.

8.2.3.1. Federation Management Proxy

The main issue regarding interoperability with other Federation Managers is that they may not be ready to participate in a "push" architecture. This is the case in this Testbed. In order to solve this, a proxy implementation containing a Resource Owner client has been developed. This proxy, "owned" by Federation Manager #2, will constantly poll endpoints listing resources and consume Federation Manager #1 API endpoints to ensure propagation of information between the two of them.

An amendment to the sequence diagram [Figure 14](#) has been made to reflect these changes:

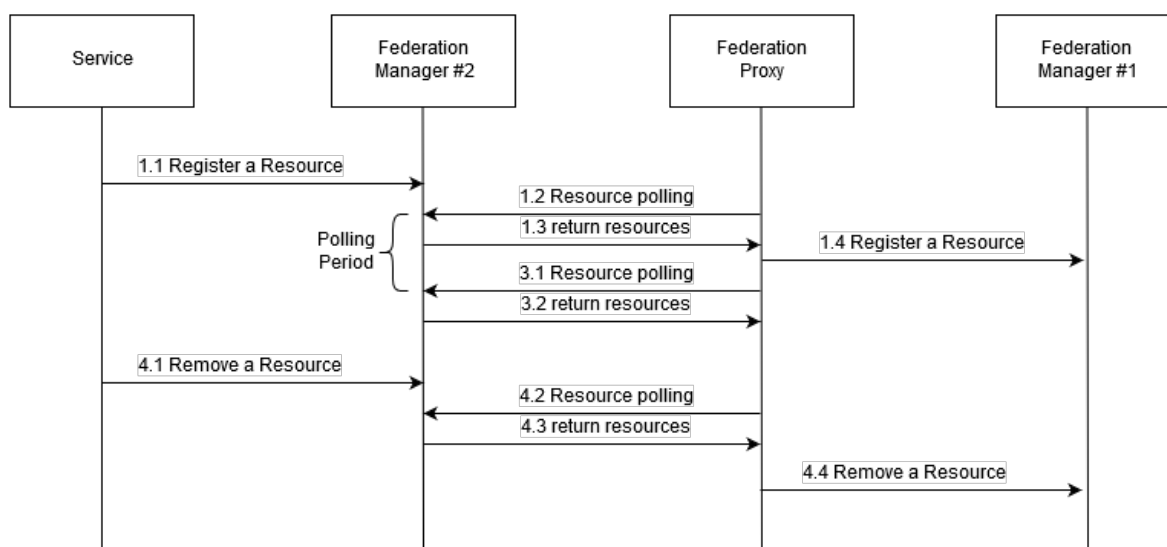


Figure 16. Resource Publishing across Federation Managers using a proxy

8.3. Analysis and Future Work

The focus of this deliverable was to protect resources that are highly tied to REST API implementations. For example, web service implementations of the emerging OGC APIs that host features, processing tools, coverage files, or any other resource made available through a REST API. The applicability to web services makes it more scalable than other resource management options such as certificate signing, which is a key factor on its viability as a Federated Clouds solution.

The API endpoints are also potentially usable in more operational scenarios where access to stored datasets needs to be protected (such as S3 interfaces that sit in front of data stores or "data lakes"). Please note both protected APIs (for Membership and Resource Management) can be leveraged in M2M (Machine-to-Machine) interactions allowing higher levels of automation and maturity of the federation. Further, this allows the creation of complex, cross-domain workflows with components that reside in independent security domains.

Future work should explore the situation whereby an external federation acts as an external resource by treating its users and resources as if they were already in the original Federation. In this case, the administrator would act as a Resource Owner, restricting the external visibility of his resources.

The final added value for this deliverable resides in the ability to allow external Federation Managers to consume these services and to at least provide a governance model that allows all

NIST deployment models to be tested [\[3\]](#).

For the purposes of Testbed-15, a minimal demonstration was made available as an ad-hoc service to the Federation Manager, giving access to as much of the functionality mentioned in this section, and the usage of personal and intermediate X.509 certificates has also been demonstrated in the Technology Integration Experiments ([TIES](#)).

Chapter 9. Federation Manager #2

9.1. Introduction

The rasdaman datacube analytics engine was the Federation Manager (FM) #2 for the Testbed-15 federation experiments. The engine uses a location-transparent distributed database. For Testbed-15, the engine was coupled with the Deimos FM #1 described above. In this section the Federation Manager #2 and its inter-federation with FM #1 are described.

9.2. Federation Definition

As Testbed-15 addressed data and service offerings (cf. definition of Data Federations and Service Federations earlier) considering existing work is adequate. Federations have been discussed for decades in Distributed Database Systems. A compact definition of a distributed database is: "*A distributed database management system (DDBMS) is a set of multiple, logically interrelated databases distributed over a network. They provide a mechanism that makes the distribution of data transparent to users.*" ([Techopedia](https://www.techopedia.com/definition/14686/distributed-database-management-system-ddbms) [https://www.techopedia.com/definition/14686/distributed-database-management-system-ddbms]) Among the "mechanisms that makes the distribution of data transparent" are means to

- Manage the logically united offering, both locally (for site administrators) and globally (for federation administrators)
- Provide a common, unified appearance to clients, such as establishing location-transparent query processing.

Such functionality is closely related to Distributed Databases, and suggests a discussion regarding similarities and differences of "federated vs distributed databases". As a starting point, the "border" between both architectures is fuzzy. Basically, a distributed database is managed by a database controller (just one), has one overarching schema, and hence forms a logical and functional unit. A federation of databases instead involves a coupling of independently existing (and evolving) databases, without centralized control.

For example, in a distributed relational database the sub-tables of a distributed table cannot change their schema (such as list of attributes) unilaterally. In a federation, data can evolve independently as each node retains full autonomy. The analysis performed during Testbed-15 found that, regarding access control, operators generally prefer local autonomy in database design and access control. However, this ER acknowledges that this approach is empirical and may not apply to all data centers and their policies.

9.3. rasdaman Architecture Overview

rasdaman is different from the FM #1 architecture in that a built-in, tightly integrated federation manager component is available. This section briefly describes the rasdaman architecture. More details and Big Earth Data application scenarios can be found in [7].

The rasdaman engine resembles a full-stack [Array DBMS](http://en.wikipedia.org/wiki/Array_DBMS) [http://en.wikipedia.org/wiki/Array_DBMS]

where each component is designed and optimized for management and retrieval of massive multi-dimensional arrays. The raster query language used by rasdaman, rasql, extends SQL with declarative n-D array operators. As such, rasql forms the blueprint for "Array SQL", [ISO SQL 9075-15:2019 Multi-Dimensional Arrays \(MDA\)](https://www.iso.org/standard/67382.html) [https://www.iso.org/standard/67382.html].

Tiled array storage can be configured for some relational database system (such as PostgreSQL), directly in some file system (which is about 2x faster than using a DBMS), or by accessing pre-existing archives without copying data into rasdaman. A versatile intelligent ETL tool homogenizes data so that within each datacube there is one common coordinate reference system, resolution (if regularly gridded), and null value set thereby also applying defaults, responding to missing data and metadata, and so forth. Likewise, internal database tuning can be performed by the rasdaman database administrator, including adaptive data partitioning ("tiling"), lossless and lossy compression, indexing, and cache sizing.

Incoming queries undergo a variety of individual optimizations, including query rewriting, common subexpression elimination, cost-based optimization, join strategy selection, multi-core parallelization, semantic caching, etc. Queries can be split over multiple nodes in a cloud for efficient parallel evaluation. By using the same principle of query splitting not only within clouds, but also across data centers, datacube federations can be established. In such a rasdaman federation, worker processes can fork subqueries to other nodes for load sharing and data transport minimization. The Figure below illustrates how expressions submitted by a client get split and dispatched according to data locations ("ship code to data"). The illustration presents a query that has been successfully distributed across more than 1,000 Amazon Web Services (AWS) cloud nodes. This is described in more detail below.

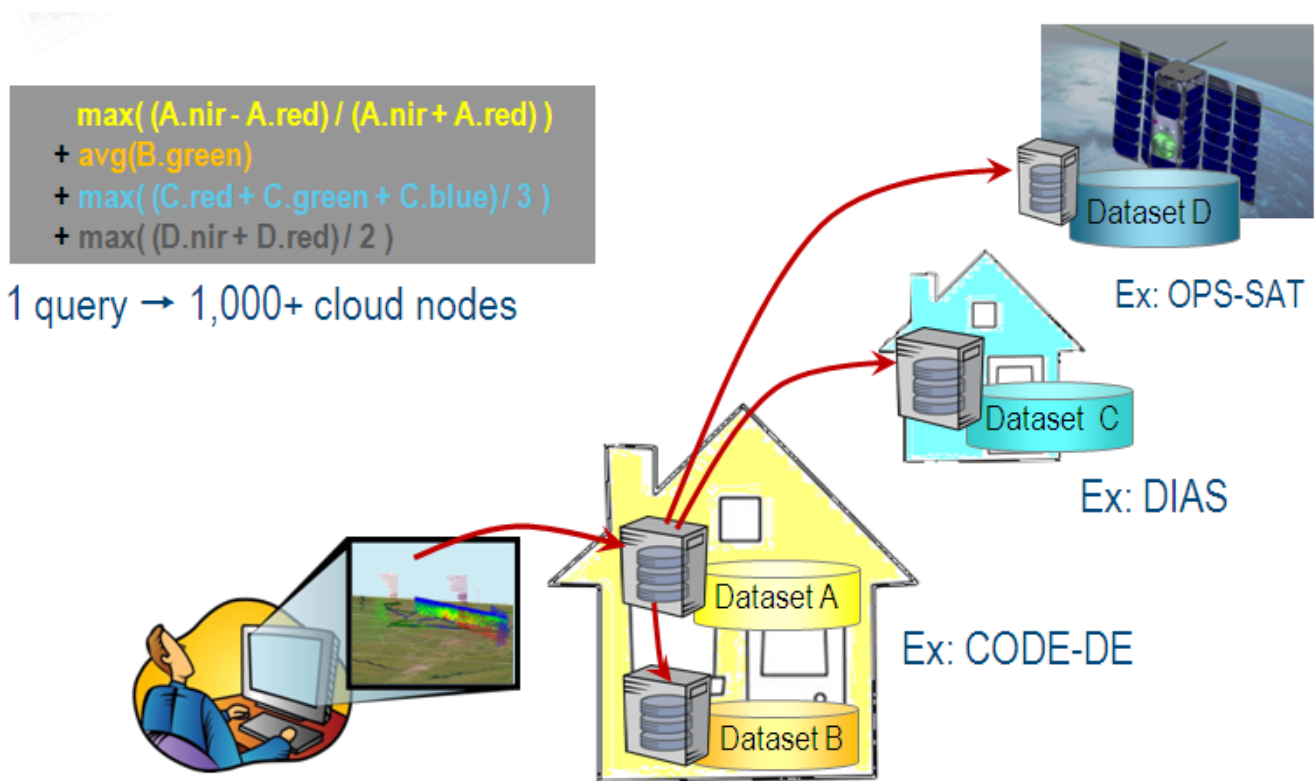


Figure 17. rasdaman Federation

The rasdaman system described so far is domain independent and can just as well serve and analyze cosmological simulations, gene expression, fintech, and business data, to name a few examples. Spatio-temporal semantics as per [OGC CIS](http://docs.opengeospatial.org/is/09-146r6/09-) [http://docs.opengeospatial.org/is/09-146r6/09-]

146r6.html] are added using an extra layer with information on coordinate reference systems (CRS) as well as regular and irregular grids, and so forth. This service interface implements OGC Web Map Service (WMS), Web Coverage Service (WCS), and Web Coverage Processing Service (WCPS). Spatio-temporal coverage functionality ranges from simple subsetting and re-formatting to complex datacube analytics.

9.4. rasdaman Federation Overview

In a rasdaman federation, the user sends a query to one of the rasdaman servers, which in turn might send subqueries to different federation partners. The rasdaman federation is location transparent. Queries can be sent to any federation member, and this member will orchestrate automatically with all members as needed (i.e. holding relevant data) to jointly process and return the query result. There is no central federation manager, rather a distributed federation mechanism is integrated with each server node. Technically, this is similar to a DNS: the network permanently updates itself and each node propagates information on changes. Hence, every node knows the global situation at any time (modulo networks latency) - knowledge that is also used for query optimization. Exchange turns out to cause negligible additional network load. Nodes still know what their local data and what remote data are, and this is communicated to users via the standard Capabilities document. The advantages of such an approach are:

- There is no single point of failure in the federation;
- All necessary autonomy is retained with each server instance (i.e. its administrators).

A basic principle of a rasdaman federation is that **Authentication is global** while **Authorization is local**. Global authentication - either via the built-in rasdaman authentication manager or some external authentication server - ensures that query fragments forwarded to federation partners will be accepted by those. Authorization is strictly local - only the local administrator can define roles, but no external entity (such as a separate global federation manager). This maximizes security while leaving enough flexibility in distributed query processing in combination with individual data policies, protecting down to the level of single pixels [8].

A federation is established by service owners who agree on mutual trust when federated queries are involved, combining own with federation member data for query answering. This is a decision made by humans, reflected by:

- A technical configuration in each rasdaman node so as to allow federation through "inpeers" (nodes from which queries are accepted - in case of open services this will usually not be restricted) and "outpeers" (nodes to which this service will send subqueries if necessary - this will typically be restricted to federation members)
- In case secure communication is required, nodes communicating must exchange public keys to allow communication encryption.
- A common understanding about user names must be established, either by establishing a common subset of users across all federation members or by agreeing on a common identity provider (or a mix of both).

The rasdaman technology allows any number of individual, independent federations, independent from each other. Likewise, it is possible that some federations merge into a larger one, or even split into independent federations.

In the extreme case, a single administrator with full power over all member nodes may determine all these settings across the complete federation. At the other end, authorization and access control remain fully local with each node, with full autonomy retained. While a coordinated approach technically is not necessary some level of coordination appears feasible in practice as the resulting network easily can become unmanageable as it grows. For example, disallowing access to objects advertised does not make much sense and should be avoided - one reason why a federation should be based on some trust and agreement among the federation operators. In the end, however, the degree of centrality vs distributed management is simply a matter of configuration. In summary, the mechanics (trust via mutually accepting the other fed members, secured communication, common identity management, etc.) in rasdaman is provided by the software, the governance model on top of this is up to the federation operators.

Establishing such governance models requires human agreements. The participants in this Testbed activity recognized that the decision whether to join a federation becomes an executive topic, involving rules & procedures that are deemed acceptable. In the emerging European Datacube Federation, a charter is being elaborated establishing common understanding and governance principles as the "human facet" of the federation. A General Assembly with one vote per data provider will decide about acceptance of new federation members. The goal is to provide governance rules which a concrete federation may accept, or adjust, or write from scratch. Of course, upon agreement of all parties with executive federation power the rules can be changed during the lifetime of the federation.

In Testbed-15, FM #2 has been instantiated with a rasdaman federation combining several AWS cloud nodes allocated in the US with nodes of the emerging European Earth Datacube Federation. Queries can be sent to any node while freely addressing and combining data from anywhere in this federation.

Based on this starting point as FM #2, in Testbed-15 a hierarchy of federations has been experimented with combining two basically "flat" federations consisting of FM #1 and #2 as a novel idea. Due to the different approaches of the heterogeneous combination (such as location-transparency vs location-awareness) the combined federation will not likely be able to offer the union of all features, but at best the intersection. In case of FM #1 and #2 this was a common authentication scheme based on a single identity manager to which FM #2 "outsources" authentication. Notably, Testbed-15 was repurposed upfront with an asymmetric approach in mind where FM #1 offers services to be consumed by FM #2, to compensate for the fact that this behavior could not necessarily be reciprocal.)

9.5. Access Control

Common role-based access control (RBAC) governs authentication and authorization. Named users are associated with roles that can be recursively nested with sub-roles. As an innovation, triggers protect datacubes from all access. A trigger is defined by the administrator as a constraint on a datacube that any query must fulfill, such as "this area within the datacube must not be accessed"

or “at most N pixels may be accessed by a query”; all triggers applying get evaluated dynamically on incoming queries acting like a guardian or a watchdog on “their” datacube. However, users and roles can get exempted from such a guardian so that queries can “pass”, thereby establishing privileges for particular users.

This mechanism establishes mandatory access control: data sets by default are protected against all users (including those getting defined in the future) and any exemption is introduced explicitly by the administrator. This way there is a guarantee that triggers defined on a data set need explicit exemption and cannot be missed accidentally, which would create security vulnerabilities.

Triggers are defined through some query expression which allows for protection of the complete cubes, sub-boxes, polygonal areas, or masks defined through other data sets or algorithmically, down to the level of single pixels.

A user of the federation is any local user of any server participating in the federation. Alternatively, this could be a user maintained by an external trusted Identity Manager which rasdaman has been configured to accept. The server managing a user’s information is called the user’s host.

9.5.1. Authentication

The identity of the user performing a query is established by the user’s host. Together with a list of federation level claims (in this case in the form of roles agreed upon by the federation partners), the user’s host provides a temporary access token which is forwarded in all subsequent subqueries. Tokens are encrypted and signed by the originating server using a private RSA key. For a server receiving an access token, this offers the following guarantees:

- The contents of a token have not been tampered with and;
- The token was generated by a trusted party.

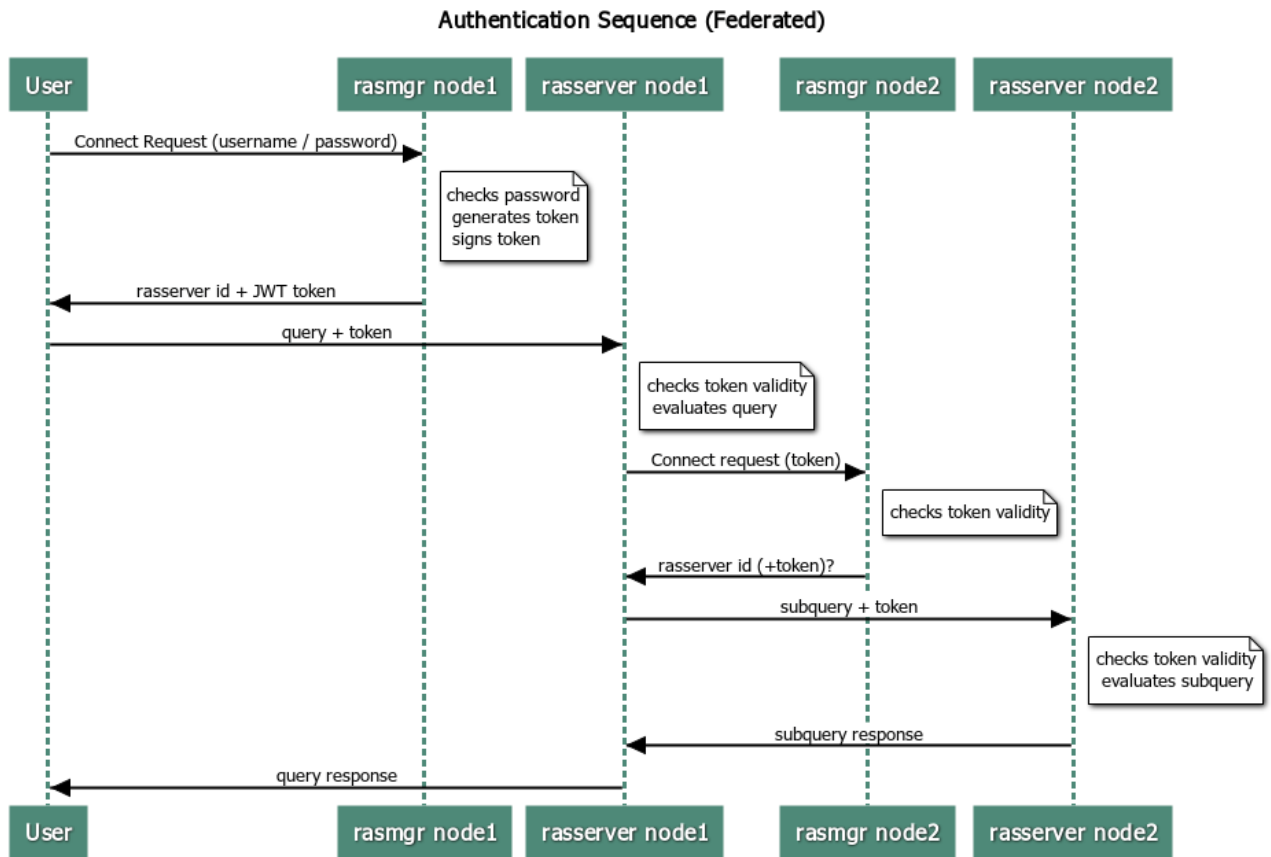


Figure 18. Federation Manager #2 Authentication

9.5.2. Trust

In order for a federation partner to accept users from other federation partners, the parties must trust each other. A federation partner (A) chooses to trust another federation partner (B) by adding its RSA public key to a list of trusted keys. Once that is done, (A) can verify that an access token was signed using the private key of (B), thus guaranteeing its authenticity.

9.5.3. Authorization

FM #2 authorization is based on well-known Role-Based Access Control (RBAC). Access rights are associated with users and roles whereby users can optionally be described through roles, and roles may recursively contain further roles.

Access rights are defined via triggers that "guard" database objects. As known in database technology, triggers consist of an event (a database retrieval or update statement), a condition to be checked whenever this event occurs, and an action to be taken in case the condition is fulfilled ("ECA rules). The general syntax for such a trigger definition is:

```

CREATE TRIGGER tr
[ SELECT | INSERT | UPDATE | DELETE ] [ ON coll1, coll2, ... ]
WHEN conditionExp
BEGIN actionExp END
  
```

In the context of massive multi-dimensional arrays, it is not sufficient to just protect the complete

object as the object may be too big. For example, ECMWF hosts hundreds of Petabytes of climate data where the long tail is free whereas the most recent two weeks must be paid for. Consequently, rasdaman allows protection of areas inside arrays, defined through bounding boxes, polygons, masks, - any query expression is admissible. The following query resembles the ECMWF situation on some object ERA5:

```
CREATE TRIGGER ForbiddenArea
WHEN some_cells( accessed( ERA5, [50:100,*,*,*,*] ) )
BEGIN exception "Insufficient rights to access specified area of MyCov." END
```

As can be seen, on fulfilment of the access criterion the query is rejected. Similarly, a quota can be established preventing excessive data access, processing, or downloads. The following statement disallows downloads exceeding 1 GB:

```
CREATE TRIGGER ResultLimit
WHEN context.resultvolume > 1000000000
BEGIN exception "Cannot download more than 1 GB at once." END
```

With the help of such rules, fine-grained mandatory access control can be established. Should some user or role not be subject to such a rule then it can be exempted. This override establishes users with fewer restrictions, such as more rights. This way, administrators have a flexible tool available for defining access policies to their data assets.

Depending on the overall policies, such administrators may be strictly local (which is usually preferred), or trigger management authority may be performed remotely, thereby enabling for example, centralized authorization management.

At runtime, authorization is performed locally on each participating node. An incoming user presents a list of claims through their access token. For example, in a scenario where partners (A) and (B) agree on federation claims OPEN and SECRET, they will internally assign corresponding roles to their respective local users. Users with host A will be able to execute queries on server B by presenting an access token issued by A. On server A, the admin decides to assign user Bob the federation role OPEN and for user Alice the federation roles OPEN and SECRET. Queries received from Bob on server B will be accompanied by tokens containing the federation claim OPEN, and the ones received from user Alice by tokens containing the federation claims OPEN and SECRET.

The authorization decision is then taken based on local rules set by the administrators of each server. For example, the admin of server A might decide that Dataset 1 is available to users presenting the OPEN claim, and Dataset 2 only available to users presenting the SECRET claim. If Dataset 1 is updated with sensitive information on a particular date, the admin can deem that the data at that particular date is only accessible to users presenting the SECRET claim (or no non-local users at all), while leaving the rest of the dataset available to users presenting the OPEN claim.

The mechanism through which authorization rules can be set up by the server administrator is integrated in the rasdaman query language and is based on the triggers and exemption mechanism described above. For each dataset the administrator can define a list of triggers. A trigger can also prohibit access to a dataset or a trigger can prohibit access from a subset of a dataset (e.g. no access

over Germany). Further, a trigger can prohibit certain operations (e.g. no interpolation over Norway), and also prohibit queries with certain properties (e.g. no queries over Germany which read more than 10GB of data, or no queries at all where the result size is more than 1TB). By default, when a trigger is defined on a dataset, all queries involving the dataset will activate the trigger. The administrator can then set up exceptions for triggers for categories of users.

9.6. rasdaman as Federation Manager #2 in Testbed-15

In Testbed-15, a rasdaman federation was established combining several AWS cloud nodes allocated in the US with nodes of the European Earth Datacube Federation containing in particular a DIAS and CODE-DE, the German Sentinel hub, providing WCS, WMS and WCPS access to Sentinel 2, Temperature and Precipitation datasets. The federation served users of both FM #1 and FM #2, and access control rules can be set for both categories. An external entity, with appropriate privileges, could manage authorization within the federation realm of FM#2, although FM#1 did not exploit this in Testbed-15. Authentication was performed via the identity provider of FM#1.

When visiting the FM #2 endpoint, the user was greeted by a log in screen, where a choice of local authentication (for the FM #2 users) and external authentication (for the FM #1 users) was available. Authentication of external users (users of FM #1 accessing data managed by FM #2) was accomplished by implementing an OAuth authentication flow against the endpoint provided by FM #1. Users choosing to authenticate using OAuth were redirected to an FM #1 page where they can log in using their FM #1 credentials. Upon successful authentication, FM #1 generated an authorization code and redirected the user back to FM #2. FM #2 then uses the authorization code for requesting an access token from FM #1.



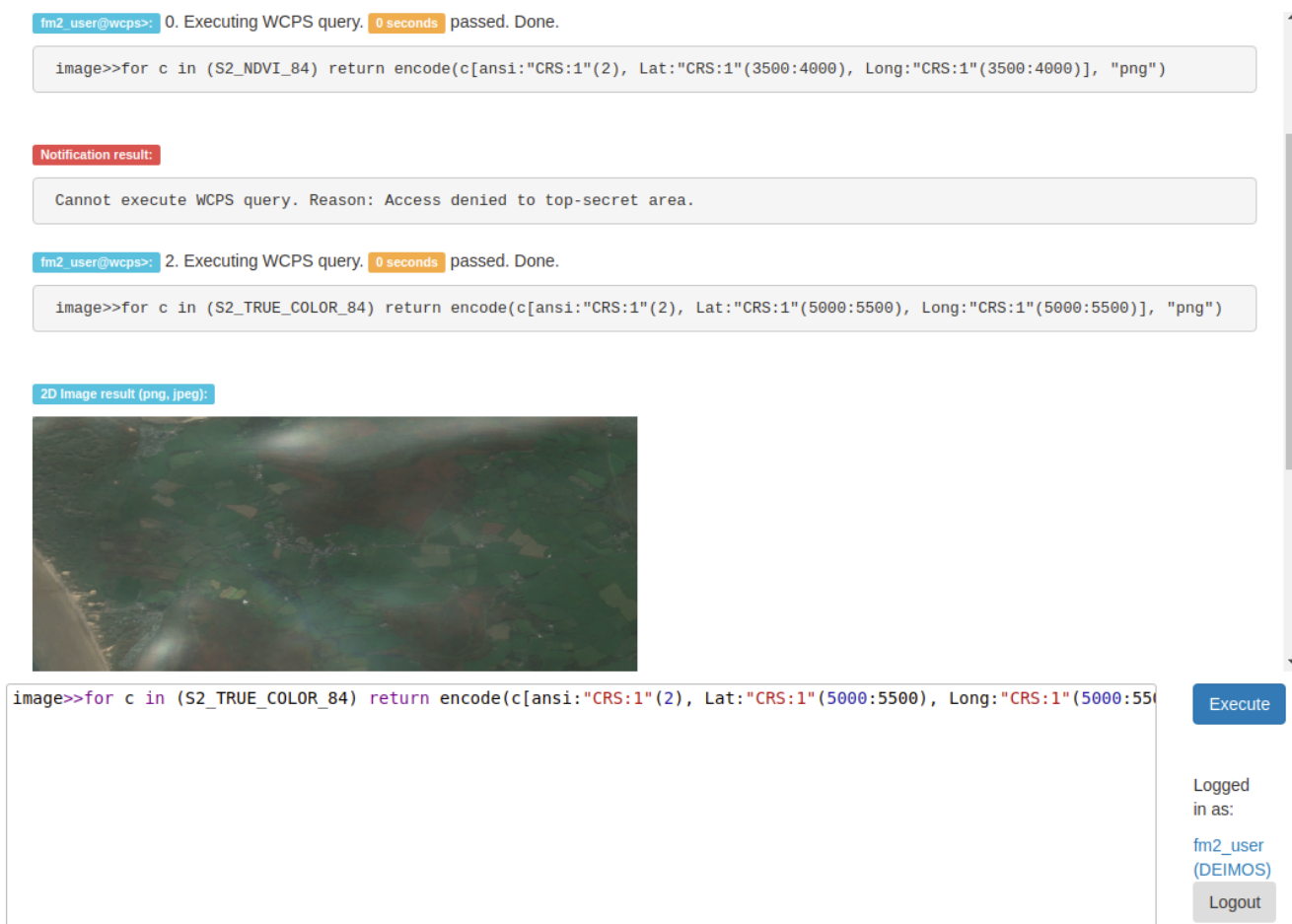
Figure 19. FM #2 log in screen. Users can use FM #2 credentials, or authenticate via FM #1.

After the authentication step, access control rules were established for both FM #1 and FM #2 users using the mechanism described in the previous section. The admin can distinguish users based on the authority issuing the access token (DEIMOS for FM #1 and rasdaman for FM #2). The admin can also assign general roles to all users issued by the same authority, as well as individual roles for

each user taking part. Then exemptions from trigger compliance preventing access in particular cases can be issued for any of these roles.

For example, consider a situation where a trigger that prevents access to a secret area of the Sentinel 2 True Color dataset was established by the FM #2 admin. An exception from the trigger is then created for the role assigned to all FM #1 users. The net result is: No user of FM #2 can access the secret area, but all users of FM #1 can. Conversely, an exception from a trigger preventing access to a secret area of Sentinel 2 NDVI was granted to the role assigned to all FM #2 users, making this area available to local users only (an FM #1 user trying to access the area would receive an Access Denied error). Further triggers and exemptions were defined based on finer grained roles of local FM #2 users. In the end, in the federation managed by FM #2, there are several types of datasets (or parts of datasets):

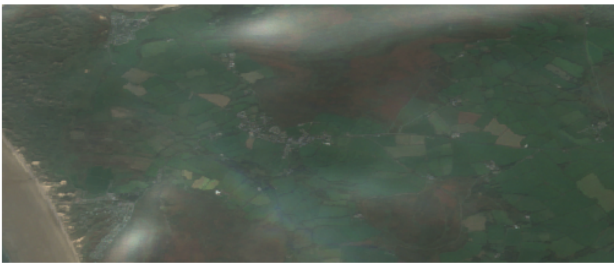
- Available to FM #2 admins only
- Available to all FM #2 users, but not to FM #1 users
- Available to FM #1 users, but not to FM #2 users,
- Open for all users.



```
fm2_user@wcps> 0. Executing WPCS query. 0 seconds passed. Done.
image>>for c in (S2_NDVI_84) return encode(c[ansi:"CRS:1"(2), Lat:"CRS:1"(3500:4000), Long:"CRS:1"(3500:4000)], "png")

Notification result:
Cannot execute WPCS query. Reason: Access denied to top-secret area.

fm2_user@wcps> 2. Executing WPCS query. 0 seconds passed. Done.
image>>for c in (S2_TRUE_COLOR_84) return encode(c[ansi:"CRS:1"(2), Lat:"CRS:1"(5000:5500), Long:"CRS:1"(5000:5500)], "png")

2D Image result (png, jpeg):


image>>for c in (S2_TRUE_COLOR_84) return encode(c[ansi:"CRS:1"(2), Lat:"CRS:1"(5000:5500), Long:"CRS:1"(5000:5500)], "png")
```

Execute

Logged in as:
fm2_user (DEIMOS)
Logout

Figure 20. FM #1 user working on FM #2 data: access to NDVI data is not allowed, access to true color data is allowed.

9.7. Analysis and Future Work

Relying on industry standards like Role-Based Access Control, SAML-based authentication, etc. allowed for a relatively straightforward connection of the rasdaman federation with FM#1, as

described. The tight integration of all components in rasdaman minimizes complexity for the administrator while offering a maximum of flexibility in policy configuration. That said, the security configuration spectrum possible, from centrally managed federations to completely autonomous federation members, may be nontrivial to oversee for federation administrators.

Consequently, future testbeds should look to define governance rules for federations, including prefabricated specializations for standard situations such as: scientific open services; scientific open services with embargos; commercial services with paid access; tightly protected services and private federations. Goal is to provide best practice solutions for establishing federations to (i) ease setup and management of federations and (ii) safely establish and maintain a defined security level. In parallel to this conceptual work an aggressive growth of the existing federations is under work, with an eye on merging them where the federation policies are sufficiently compatible.

Overall, the diversity of federation approaches in Testbed-15 has surfaced important issues on service federations in general and has led to initial insights. Insofar, Testbed-15 has kicked off charting the new field of federation management.

Chapter 10. Technology Integration Experiments

10.1. Components

This section contains the list of components that participated in TIE execution during Testbed-15.

Name	Deliverable ID	Endpoint Samples
Federation Manager #1	D143	https://testbed15-fm.elecnor-deimos.com/.well-known/openid-configuration https://testbed15-fm.elecnor-deimos.com/swagger-ui/
Federation Manager #2	D144	http://34.239.178.81:8080/rasdaman/ows
Services (W*S)	D147	http://34.239.178.81:8080/rasdaman/ows
FM Proxy	-	-
CMD Client	-	-

It is important to take into account that due to the fact that D144 and D147 are highly coupled:

- Pairings between both FM#1 and FM#2 are indirectly also involving W*S Services.
- An in-kind component "FM Proxy" was introduced to the list of deliverables in order to provide interoperability between Federation Managers.

10.2. Use Cases

The following matrix focuses on the use cases enabled by each of the Deliverables. These use cases are extensively defined in [management](#), and server to refine a final TIE list with viable functionality tests.

The tests could only partially rely on OGC W*S services as not all FMs offer interfaces accessible through default clients. FM #1 requires a separate access step whereas FM #2, due to its integrated federation manager, allows standard clients like Leaflet, ESA-NASA WebWorldWind, python, and R to use the vanilla OGC WMS, WCS, and WCPS functionality.

Code	Provided functionality	FM#1	FM#2
W1	Allow standard WMS access	No	Yes
W2	Allow standard WCS access	No	Yes

Code	Provided functionality	FM#1	FM#2
W3	Allow standard WCPS access	No	Yes

Code	Provided functionality	FM#1	FM#2
Internal within the Federation			
I1	Provide Authentication to SPs	Yes	Yes
I2	Allow discoverability by Client Applications	Yes	Yes
I3	Allow access to unprotected resources	Yes	Yes
I4	Allow transparent access to resources	No	Yes
I5	Act as PDP during resource access	Yes	Yes
I6	Allow protection of resources	Yes	Yes
I7	Grant and Revoke Access	Yes	Yes
External to the Federation			
E1a	Provide Authentication to external FM	Yes	No
E1b	Use external IDPs for authentication	Yes	Yes
E2a	Discover external FM resources (pulling)	Yes	No
E2b	Discoverability from external FM	Yes	No
E3a	Allow use of external PDP (other FMs)	Yes	No
E3b	Allow use as PDP from external SPs	Yes	No
E4a	Grant-revoke access in external FM	Yes	Yes(*)

Code	Provided functionality	FM#1	FM#2
E4b	Allow grant-revoke access from external FM	Yes	No

*Provided by the FM Proxy component

The table shows two Federation Manager components able to provide their federation environments with all the internal functionality identified in this ER. Nevertheless, they fail to fulfill the purpose of inter-federation with each other due to several reasons:

- FM#2 does provide integrated federation, but does not provide any Identity Provider, Policy Decision Point and Federation Management interfaces to components outside of its own Federation.
- FM#2 does not consume external PDP endpoints for Authorization purposes. FM#2 does offer a declarative interface for managing authorization through an instance such as FM#1, but FM#1 did not utilize this interface in this Testbed.
- FM#1 does not support interaction with dynamic service orchestration standards to the extent FM#2 does, hence FM#1 could not use the service interfaces for defining authentication.
- FM#1 expects to be part of a "push" architecture expecting requests on its particular API whereas FM#2 operates solely on the OGC W*S architecture (expecting users to "pull" information from previously known service endpoints), although it allows dynamic queries regarding discoverability of resources.

Following the strategy defined in the Testbed-15 roadmap a "one-way" Federation was established where FM#2 connects with the authentication functionality provided by FM#1, and to provide an approach to solve the differing approaches between both that facilitates future TIEs that aim at tackling the rest of use cases not covered in the pairings of this Testbed, assessing the various options and facets elaborated in this experiment, and ultimately advancing interoperability of federations.

10.3. Pairings

The following TIE tests have been performed:

Components	Use Case	Result
FM#1 / FM#2	FM#1 propagates authentication information to FM#2	OK
FM#1 - FM Proxy / FM#2	FM#1 consumes service metadata from FM#2 and populates its resource catalogue	OK
FM#2 / FM#1	FM#2 propagates authentication information to FM#1	NOT OK

Components	Use Case	Result
CMD Client / FM#1	FM#2 administrator updates user privileges on FM#1 membership service	OK
CMD Client / FM#2	FM#1 administrator updates user privileges on FM#2 membership service	OK
FM#2 / FM#1	FM#2 systematically updates user privileges on FM#1 membership service	NOT OK
FM#1 / FM#2	FM#1 systematically updates user privileges on FM#2 membership service	NOT OK

10.4. Experiments

10.4.1. FM#1 propagates authentication information to FM#2

End-Users with an already existing account in FM#1 can use their credentials to access FM#2 services. This test allows to verify that FM#2 is able to delegate its authentication capabilities to an external Identity Provider, and the ability of FM#1 to provide Federated Membership Management capabilities outside of its Federation environment.

10.4.2. FM#1 consumes service metadata from FM#2 and populates its resource catalogue

An ad-hoc proxy in FM#1 domain maintains an updated list of FM#2 resources. This test allows to verify that FM#1 can received and register references to external W*S services belonging to FM#2.

10.4.3. FM#2 propagates authentication information to FM#1

End-Users with an already existing account in FM#2 can use their credentials to access FM#1 services, since FM#2 only provides IDP functionality to services within its own federation.

10.4.4. FM#2 administrator updates user privileges on FM#1 membership service

Through calls to the FM API provided by FM#1, this test allows to verify Federated Access and Membership Management by actors external to FM#1 that belong to FM#2.

10.4.5. FM#1 administrator updates user privileges on FM#2 membership service

FM#2 offers a declarative interface which allows after proper authentication the definition, manipulation, and retrieval of authorization information.

10.4.6. One FM systematically updates user privileges on other FM membership service

While on principle it seems possible that one FM can utilize the API of the other FM for both extracting and propagating authorization information the API, this behavior was not exercised. A FM Proxy was put in place to mitigate this functionality gap.

10.5. Summary

Two federation managers with differing approaches have been inspected and paired in Testbed-15. FM#1 is a standalone component offering authentication and authorization to which nodes can subscribe; its service remains visible to clients which need to approach FM#1 first, and then approaches the target service as usual. FM#2 is tightly integrated in a fully distributed, location-transparent service, and clients accessing the service do not need to perform an extra step when invoking the federation service. Both architectures lead to different technical consequences which have been investigated and described in this report. As such, the report forms a suitable basis for further investigation on the interoperability between and across federations.

Appendix A: Revision History

Table 1. Revision History

Date	Editor	Release	Primary clauses modified	Descriptions
May 23, 2019	H. Rodriguez	.1	All	Initial ER.
Aug. 16, 2019	H. Rodriguez	.2	All	Partial DER (pending section 8).
Sep. 13, 2019	C. Reed	.2	All	Internal review completed.
Oct. 03, 2019	H. Rodriguez	.3	All	Near-Final DER.
Oct. 29, 2019	H. Rodriguez	.4	All	Final DER (r1)
Dec. 16, 2019	H. Rodriguez	.5	All	Final Edits (r2)
Dec. 18, 2019	G. Hobona	.5	All	Final staff edits.

Appendix B: Bibliography

1. OGC: OGC Web Services Security, <http://docs.opengeospatial.org/is/17-007r1/17-007r1.html>.
2. The EGI Federation, <https://www.egi.eu/federation/>.
3. The NIST Cloud Federation Reference Architecture, https://www.nist.gov/sites/default/files/documents/2019/07/09/nist_cfra_20190709_draft_v1.0.pdf.
4. Sampath, A., Tripti, C.: Synchronization in Distributed Systems. In: Meghanathan, N., Nagamalai, D., and Chaki, N. (eds.) *Advances in Computing and Information Technology. Advances in Intelligent Systems and Computing*, vol 176. Springer, Berlin, Heidelberg (2012).
5. Doval, J.J., Rodríguez, H.: OGC Testbed-14: Security Engineering Report. OGC 18-026r1, Open Geospatial Consortium, <https://docs.opengeospatial.org/per/18-026r1.html> (2018).
6. Lee, C.A.: OGC Testbed-14: Federated Clouds Engineering Report. OGC 18-090r1, Open Geospatial Consortium, <https://docs.opengeospatial.org/per/18-090r1.html> (2019).
7. Baumann, P., Rossi, A.P., Bell, B., Clements, O., Evans, B., Hoenig, H., Hogan, P., Kakaletris, G., Koltsida, P., Mantovani, S., Figuera, R.M., Merticariu, V., Misev, D., Huu, B.P., Siemen, S., Wagemann, J.: *Fostering Cross-Disciplinary Earth Science Through Datacube Analytics*. In: P.P. Mathieu, C.A. (ed.) *Earth Observation Open Science and Innovation - Changing the World One Pixel at a Time* pp. 91 - 119. International Space Science Institute (ISSI) (2017).
8. Baumann, P., Misev, D.: *Access Control on Big Data and Small Pixels: How to Achieve Privacy and Security*. IEEE International Geoscience and Remote Sensing Symposium. July 28 - August 2, 2019.