



D3.4 - Data Integrity and Security Assurance in the RTM architecture

WP3: Real-Time Water Efficiency Monitoring Platform for the Process Industry

June 2023

Authors: Evangelia Anagnostopoulou (ICCS), Dimitris Apostolou (ICCS), Babis Magoutas (ICCS), Efthimios Bothos (ICCS), Dimitra Pournara (ICCS), Gregoris Mentzas (ICCS), Aitor Corchero Rodriguez (EUT), Robert Sanfeliu Prat (EUT), Nele Desmet (VITO), Janelcy Alferes Castano (VITO), Kostas Kalaboukas (MAG), Aziz Mousas (MAG)



The AquaSPICE project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 958396.

Document Information

GRANT AGREEMENT NUMBER	958396	ACRONYM	AquaSPICE
FULL TITLE	Advancing Sustainability of Process Industries through Digital and Circular Water Use Innovations		
START DATE	1 st December 2020	DURATION	48 months
PROJECT URL	www.AquaSPICE.eu		
DELIVERABLE	D3.4 – Data Integrity and Security Assurance in the RTM architecture		
WORK PACKAGE	WP3 – Real-Time Water Efficiency Monitoring Platform for the Process Industry		
DATE OF DELIVERY	CONTRACTUAL	11/2022	ACTUAL 06/2023
NATURE	Demonstrator	DISSEMINATION LEVEL	Public
LEAD BENEFICIARY	ICCS		
RESPONSIBLE AUTHOR	Evangelia Anagnostopoulou (ICCS)		
CONTRIBUTIONS FROM	Dimitris Apostolou (ICCS), Dimitra Pournara (ICCS), Gregoris Mentzas (ICCS), Babis Magoutas (ICCS), Efthimios Bothos (ICCS), Aitor Corchero Rodriguez (EUT), Robert Sanfeliu Prat (EUT), Janelcy Alferes Castano (VITO), Kostas Kalaboukas (MAG), Aziz Mousas (MAG)		
ABSTRACT	This deliverable investigates the security requirements to be considered in order to ensure the security of water data within the AquaSPICE framework. It describes the development of a policy model for formally describing flexible context-based access control and consent rulesets, as well as static policies concerning the manner in which AquaSPICE data (e.g. water efficiency system's data) can be used.		

Document History

VERSION	ISSUE DATE	DESCRIPTION	CONTRIBUTOR
0.1	01/04/2022	Table of Contents shared with partners	ICCS
0.4	05/09/2022	Integrated security requirements from CPS partners	ICCS
0.8	24/10/2022	Final draft sent for internal review	MAG
1.0	30/11/2022	Final version to be submitted	ICCS
1.1	30/04/2023	Revised version considering CS#4 as voice case	ICCS

Disclaimer

Any dissemination of results reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

Copyright message

© AquaSPICE Consortium, 2022-2023

This deliverable contains original unpublished work except where indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation, or both. Reproduction is authorised if the source is acknowledged.

TABLE OF CONTENTS

- 1. Executive summary..... 7
- 2. Introduction..... 8
- 3. State of the Art 9
 - 3.1. ABAC Architecture..... 9
 - 3.2. Context in ABAC..... 10
- 4. AquaSPICE Security Requirements 12
 - 4.1. Security requirements for context-aware access policies..... 12
 - 4.1.1. Security requirements for Case Study #1 (DOW)..... 14
 - 4.1.2. Security requirements for Case Study #2 (UNIVPM)..... 14
 - 4.1.3. Security requirements for Case Study #4..... 18
 - 4.1.4. Security requirements for Case Study #5 (Agricola)..... 18
 - 4.1.5. Security requirements for Case Study #6 (Tupras)..... 19
- 5. AquaSPICE Approach 24
 - 5.1. RTM Architecture..... 24
 - 5.2. Data Integrity 24
 - 5.3. Security Provider..... 25
 - 5.4. Securing Access..... 27
 - 5.4.1. Keyrock Identity Management..... 27
 - 5.4.2. PDP Access control..... 28
 - 5.5. Manage Roles Tutorial 30
 - 5.6. Demo Example 33
- 6. Conclusions and Next Steps..... 35
- 7. Appendix: Security Questionnaires & Technical Instructions 36
 - 7.1. Questionnaire for security requirements for context-aware access policies 36
 - 7.2. Instructions 37
- 8. References..... 39

LIST OF FIGURES

- Figure 1: ABAC Architecture (source: <https://py-abac.readthedocs.io/en/latest/concepts.html>) 9
- Figure 2: RTM Architecture..... 24
- Figure 3: Grafana login page..... 26

Figure 4: Login page of the Keyrock Identity Manager	27
Figure 5: Example of a simple access rule	30
Figure 6 Keyrock - roles to permissions mapping	31
Figure 7: Create roles and permissions.....	31
Figure 8: Create a role	32
Figure 9: Create permission.....	32
Figure 10: Create advanced XACML Rule.....	33
Figure 11: Keyrock - New role and assignments	33

LIST OF TABLES

Table 1: Overview of security requirements for context-aware access policies	14
--	----

ABBREVIATIONS/ACRONYMS

DT	Digital Twin
RTM	Real Time Monitoring
XACML	eXtensible Access Control Markup Language
PEP	Policy Enforcement Point
PDP	Policy Decision Point
PIP	Policy Information Point
PAP	Policy Administration Point
ABAC	Attribute-based Access Control
OPA	Open Policy Agent
CapBAC	Capability-based Access Control
CS	Case Study
CPS	Case Pilot Study
SSO	Single Sign-On

1. Executive summary

This deliverable investigates the security requirements to be considered in order to ensure the security of water data within the AquaSPICE framework. It describes the development of a policy model for formally describing flexible context-based access control and consent rulesets. It also addresses static policies concerning the manner in which AquaSPICE data (e.g. water efficiency system's data) can be used.

This deliverable also reports on the state-of-the-art security approaches for context-aware access policies. It describes the security requirements for context-aware access policies of AquaSPICE. The deliverable describes the proposed security provider service, explains how to configure rules and define permissions using the Keyrock system and presents a demo example that shows the interaction with smart devices.

Due to the non-existence of the SynDi plant in CS#4, this CS is considered as void case and no work related to this CS is reported in this deliverable.

2. Introduction

This deliverable reports on the outcomes of task T3.4 “Context-Aware Access Control, Data Integrity and Security Assurance Mechanisms”. The task investigated the main security requirements to be considered in order to ensure the security of water data within the AquaSPICE framework. It presents the development of a policy model that formally describes flexible context-based access control and consent rulesets. It also addresses static policies concerning the usage of AquaSPICE data (e.g. water efficiency system's data).

The deliverable describes the security requirements and architecture for the development of the main context-aware access policies of AquaSPICE. The deliverable takes into account the proposed architecture of the real time monitoring platform for AquaSPICE project which is the outcome of Task 3.1 “RTM System Architecture”, reported in Deliverable 3.1 “RTM Specification and System Architecture.” This deliverable also reports on the state-of-the-art security approaches for context-aware access policies, analysing prominent choices and making recommendations for adoption and use in AquaSPICE. The deliverable describes the proposed security provider service, explains how to configure rules and define permissions using Keyrock and presents a demo example that shows the interaction with smart devices.

The results presented herein will be used in Task 3.5 “Real-Time Water Monitoring Platform”, which will develop the RTM platform.

3. State of the Art

3.1. ABAC Architecture

The ABAC architecture, as illustrated in Figure 1, consists of several key components. These components include:

1. Policy Enforcement Point (PEP): This component is responsible for securing applications and data. It intercepts incoming requests and forwards authorization requests to the Policy Decision Point (PDP) for evaluation.
2. Policy Information Point (PIP): The PIP acts as a bridge between the ABAC system and external sources of attributes, such as LDAP databases. It retrieves attribute values that are necessary for the evaluation of access control policies.
3. Policy Administration Point (PAP): The PAP is responsible for managing access control policies. It handles the creation, modification, and deletion of policies that govern access to sensitive data.

In the ABAC model, policies are statements that combine attributes to define acceptable or denied actions, ultimately determining whether access to sensitive data should be permitted or denied. When a requestor seeks access to a specific data record, the PEP intercepts the request and sends a decision request to the PDP. The PDP then evaluates the relevant policies managed by the PAP, utilizing attribute values fetched from the PIP. ABAC has been successfully employed in controlling access to Electronic Health Record Systems, as highlighted in reference [1].

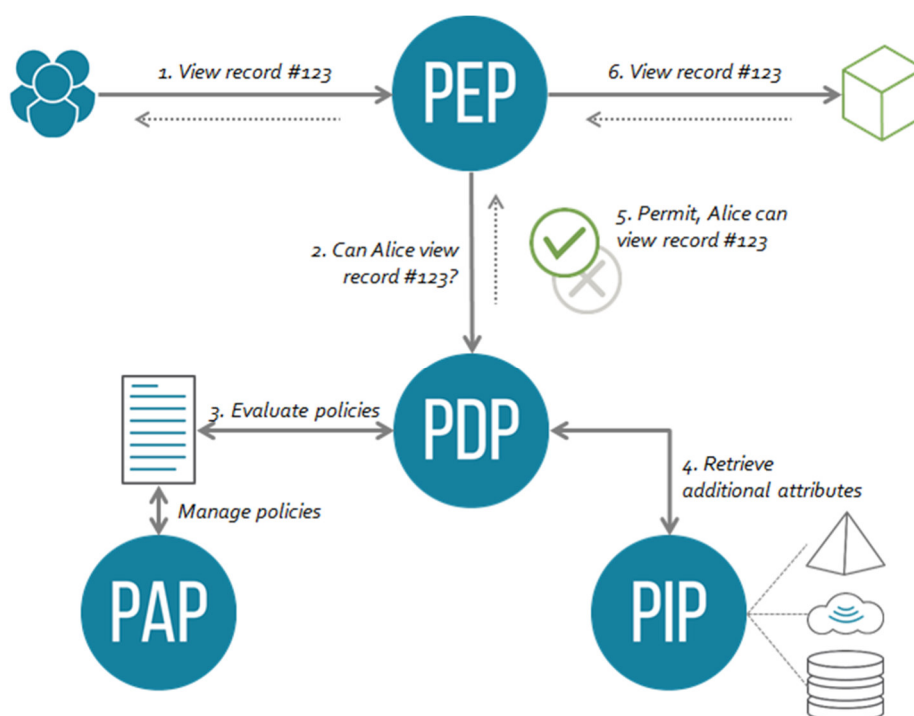


Figure 1: ABAC Architecture (source: <https://py-abac.readthedocs.io/en/latest/concepts.html>)

3.2. Context in ABAC

In the context of access control, the XACML (eXtensible Access Control Markup Language) standard [2] defines a context handler as a system entity responsible for converting decision requests from the native format to the XACML canonical form, and vice versa, converting authorization decisions from the XACML canonical form to the native response format [3]. In ABAC, context handlers play a crucial role in transforming attributes into formats that are applicable and meaningful within the application. They enable the inference of higher-level context from low-level context, which is essential for identifying critical situations, such as emergencies in industrial plants, and making access control decisions based on them.

In contrast to the XACML architecture, an alternative approach is offered by the Open Policy Agent (OPA) [4]. OPA is an open-source policy engine designed to unify policy enforcement across various systems. It presents a high-level language that allows policies to be specified as code, along with APIs that enable software to offload policy decision-making to OPA. When software requires policy decisions, it can query OPA and provide structured data as input. The expression of OPA policies is accomplished through the use of Rego, a language that is purpose-built for defining policies over intricate hierarchical data structures. It's worth noting that OPA has its own policy grammar, as mentioned by Siebach [5]. However, one challenge is that it may transcend domain boundaries when obtaining attributes, or the policy language used may not be user-friendly enough for business owners to write policies directly. For instance, the Rego language is highly expressive but may require technical expertise to write policies due to its grammar intricacies.

In practical scenarios, context handlers are specialized software components employed to process raw contextual data that is pertinent to access control decisions. Their primary role is to transform this data into instances of a context model, thereby semantically enriching it. Context handlers have the responsibility of integrating context-aware policy enforcement mechanisms with contextual information transforming it into a format that can be effectively used to evaluate access control policies. It's important to acknowledge that context handlers can be extensive, which emphasizes the need for their design and development to be informed by a suitable context model.

The necessity for context handlers that can effectively process raw contextual data and deriving valuable insights for access control was initially introduced in our previous work [6] within the cloud platform-as-a-service security domain. Our focus has been on developing context handlers with two key capabilities: i) offering real-time measurements for specific contextual attributes, and ii) elevating the registered attribute value(s) to a semantic level suitable for the application domain and the corresponding access control policy. In this current work, we are expanding and enhancing our approach to enable context-aware access control within the industrial automation domain.

Moreover, the Capability-based Access Control (CapBAC) is an alternative approach where the concept of capability, defined as a token or key granting access permission to an entity in a computer system, was initially introduced [7]. As explained by Gusmeroli et al. [8], a capability is a token of authority that uniquely references an object along with its associated access rights. When comparing ABAC with CapBAC, several differences can be observed.

ABAC offers advantages over CapBAC in the following ways: Firstly, ABAC specifies access policies by utilizing subject's properties, resource attributes, and environmental properties, resulting in more complex rules along with increased processing and data availability requirements [8]. Additionally, capability-based authorization in CapBAC necessitates issuing capabilities to all subjects and requires the requesting subject to select a capability when making a request, which can be considered a disadvantage [8]. While capability-based methods have been employed in access control solutions for IoT applications, implementing the original concept of capability-based access control in an IoT network has raised issues such as capability propagation [9]. Furthermore, in open, cross-domain, or cross-enterprise contexts, standardization of the structure of capability tokens, CapBAC supporting services, and their access protocols is crucial [8].

However, CapBAC offers advantages over ABAC as highlighted by Gusmeroli et al. [8]: Firstly, CapBAC requires a consistent attribute definition within a domain, which is considered a prerequisite for ABAC. Additionally, ABAC and RBAC systems lack flexible delegation rights features, which are supported by CapBAC.

4. AquaSPICE Security Requirements

In this section, we present the main security requirements to be considered in order to ensure the security of water data within the AquaSPICE framework. AquaSPICE's innovations emanate from the requirements of Case Studies. The security requirements presented in the following section derived from the results of the interviews conducted with domain experts. This collection of security requirements has been carried out through several activities:

- Questionnaires: A data template was shared among CPS partners, which was used to identify, analyze and gather technical and security requirements that will guide the development of the main context-aware access policies of AquaSPICE. This process led to a record of several context-aware access policies. The template that was used is reported in the Appendix, whereas the analysis of the security requirements gathered is provided next.
- CS workshops: As part of WP3, several rounds of technical workshops have been conducted for each CPS component. These workshops, where technical leaders of the CPS and partners of AquaSPICE attended, have been centred at discussing the water treatment processes to be implemented during the project and analysing the necessities with respect to the security requirements for context-aware access policies.
- Bi-lateral meetings: Apart from the questionnaire and individual workshops, several bi-lateral meetings between WP3 members and CPS leaders have been conducted to clarify the necessities regarding security requirements for context-aware access policies.

As a result of the activities, this section gives a detailed description of the collected security requirements for context-aware access policies in AquaSPICE. The security requirements were identified through a structured process, in collaboration with the partners who are responsible for each pilot. More specifically, a template was defined asking different partners to provide technical and security requirements that will guide the development of the main context-aware access policies of AquaSPICE. The process was initiated as part of the technical architecture discussions and was extended to be able to gather further information related to the project's security needs. The template was distributed as a google doc and the information that partners provided is the basis for identifying the security requirements for context-aware access policies.

In the following sections a description of the identified security requirements for context-aware access policies is provided.

4.1. Security requirements for context-aware access policies

This section describes the identified security requirements for context-aware access policies per Case Study.

	Requestor	Policy	Resources	Context
CS#1 (DOW)	Plant engineer	Access for modifications by plant engineer manager of his/her device data.	Any data to which the plant engineer manager should have access.	Requesting device owned by plant
CS#2 (UNIVPM)	Engineer or operator	Access to historical process data Access for modifications to set points for process variables/alarm limits/PID controls/advanced process controls	Any process data to which the person should have access.	Requesting device is work computers of engineers or plant computers
CS#3 (VITO)	Engineer, operator, R&D engineer or IT developer	Access to real-time and historical data of the sensor network Access to device data	Sensor measurement data Sensor device data	Requesting device is work computers of engineers or plant computers or server (machine to machine)
CS#5 (Agricola)	IT manager, Project manager, data controller, technician	Access for modifications by project manager of rented device data	Any data to which the project manager should have access	Accessible device, rented device
CS#6 (Tupras)	Engineer, operator	Access to historical process data Access for modifications to set points for process variables/alarm limits/PID	Any process data to which the person should have access.	Requesting device is work computers of engineers or plant computers

		controls/advanced process controls		
--	--	------------------------------------	--	--

Table 1: Overview of security requirements for context-aware access policies

4.1.1. Security requirements for Case Study #1 (DOW)

Policy Id & Name	P1. Access for modifications by plant engineer manager of his/her device data.				
Objectives	Plant engineer manager can access and interact with the devices of his assets.				
Resources/Scope	Any data to which the plant engineer manager should have access.				
<p>The plant engineer manager requests access to a device’s data in a read/write fashion.</p> <p>The procedure is as follows:</p> <ul style="list-style-type: none"> The plant / improvement engineer manager requests access to a device (e.g., an asset, IoT sensor, etc.). After proper identification by the data controller, the data is presented to the plant engineer manager. <p>Constraints:</p> <ul style="list-style-type: none"> Access by authorized personnel ONLY AND Read/Write Access by the plant / improvement engineer manager ONLY Access by the plant / improvement engineer manager only, for which the plant engineer MUST BE properly identified. 					
	Explicit Access Attempt/Request info			Context Conditions	
#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny
1.	Plant engineer	READ/WRITE	Device data	Device belongs to the process plant under the plant engineer manager’s responsibility	Permit
2.	Improvement engineer	READ/WRITE	Device data	Device belongs to the process plant under the plant engineer manager’s responsibility	DENY
Rule Combining Algorithm	Permit Overrides				

4.1.2. Security requirements for Case Study #2 (UNIVPM)

Policy Id & Name	P1. Access to historical process data
-----------------------------	---------------------------------------

Objectives	Any engineer or operator can access the historical process data.				
Resources/Scope	Any process data to which the person should have access.				
	Explicit Access Attempt/Request info			Context Conditions	
#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny
1.	Any engineer and operator	READ	Process Historian	From the work computers of engineers and from plant computers.	Permit
Rule Combining Algorithm	Permit Overrides				

Policy Id & Name	P2. Access for modifications to set points for process variables				
Objectives	Process control engineers and chief plant operators can access and change/override the set points				
Resources/Scope	Manipulated process values to which the process control engineers and chief plant operators should have access.				
	Explicit Access Attempt/Request info			Context Conditions	
#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny
1.	ANY	READ	Process Historian	From the work computers of engineers.	Permit
2.	ANY	WRITE	Process Control tool	From the work computers of engineers.	DENY
3.	Process control engineers & chief plant operators	READ/ WRITE	Process Control tool	From the plant computers.	Permit
Rule Combining Algorithm	Permit Overrides				

Policy Id & Name	P3. Access for modifications to process alarm limits				
Objectives	Process control engineers and chief plant operators can access and change/override/update the alarm limits of process variables				
Resources/Scope	Alarm upper and lower limit values to which the process control engineers and chief plant operators should have access.				
	Explicit Access Attempt/Request info			Context Conditions	
#	Requestor	Action	Resource	<u>Contextual Attributes</u> <i>Operator Parameters</i> AND/OR	Permit / Deny
1.	ANY	READ	Process Historian	From the work computers of engineers.	Permit
2.	ANY	WRITE	Process Control tool	From the work computers of engineers.	DENY
3.	Process control engineers & chief plant operators	READ/ WRITE	Process Control tool	From the plant computers.	Permit
Rule Combining Algorithm	Permit Overrides				

Policy Id & Name	P4. Access to simple PID control logics				
Objectives	Process engineers can access and define rules for process control				
Resources/Scope	PID control logics which process engineers should have access.				
	Explicit Access Attempt/Request info			Context Conditions	

#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny
1.	ANY	READ	Process Historian	From the work computers of engineers.	Permit
2.	ANY	WRITE	Process Control tool	From the work computers of engineers.	DENY
3.	Process control engineers	READ/ WRITE	Process Control tool	From the computers.	Permit
Rule Combining Algorithm		Permit Overrides			

Policy Id & Name	P5. Access to modifications to advance process control (model predictive control) logics				
Objectives	Process engineers can access and define rules for advance process control				
Resources/ Scope	MPC control logics which process engineers should have access.				
	Explicit Access Attempt/Request info			Context Conditions	
#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny
1.	ANY	READ/ WRITE	Process Control tool	From the work computers of engineers.	DENY
2.	Process control engineers	READ/ WRITE	Process Control tool	From the work computers of engineers and from plant computers.	Permit
Rule Combining Algorithm		Permit Overrides			

Security requirements for Case Study #3 (water-link, BASF)

Policy Id & Name	P1. Access to real-time and historical sensor measurement data				
Objectives	Any authorized engineer or operator and any authorized R&D engineer or IT developer can access the historical sensor measurement data of the AquaSPICE sensor network in Antwerp harbor and on Albert Canal.				
Resources/Scope	Sensor measurement data (temperature, conductivity and water depth)				
	Explicit Access Attempt/Request info			Context Conditions	
#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny
1.	Any authorized engineer and operator	READ	Historical data storage Context-Broker Dashboarding tool (Grafana)	From the work computers of engineers and from plant computers.	Permit
2.	Any authorized R&D engineer and IT developer	READ/ WRITE	Historical data storage Context-Broker Dashboarding tool (Grafana)	From the work computers of engineers and from server (machine to machine).	Permit
Rule Combining Algorithm	Permit Overrides				

4.1.3. Security requirements for Case Study #4

Due to the non-existence of the SynDi plant in Case Study #4, this CS is considered as void case and no work related to this CS is reported in this deliverable.

4.1.4. Security requirements for Case Study #5 (Agricola)

Policy Id & Name	P1. Access for modifications by project manager of rented device data.
Objectives	Project manager and IT manager can access and interact with the devices.

Resources/Scope	Any data to which the project manager should have access.					
<p>The project manager requests access to a device's data in a read/write way.</p> <p>The procedure is as follows:</p> <ul style="list-style-type: none"> The project manager requests access to a device (e.g. an asset, IoT sensor, etc.). After proper identification by the data controller, the data is presented to the project manager. The technician checks the data. <p>Constraints:</p> <ul style="list-style-type: none"> Access by authorized personnel ONLY AND Read/Write Access by the project manager and IT manager ONLY Access by the project and IT manager only, for which the data controller MUST BE properly identified. 						
	Explicit Access Attempt/Request info			Context Conditions		
#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny	
1.	IT manager	READ/WRITE	Device data	Can access and interact with the device	Permit	
2.	Project manager	READ/WRITE	Device data	Device is rented under the project manager's responsibility. Project manager can access and interact with the device.	Permit	
3.	Data controller	READ/WRITE	Device data	Data controller check and collect the data in scope of the project, inform project manager for any non-conformity observed.	Permit	
4.	Technician	READ	Device data	Technician check the data and inform the plant engineer	Permit	
Rule Combining Algorithm	Permit Overrides					

4.1.5. Security requirements for Case Study #6 (Tupras)

Policy Id & Name	P1. Access to historical process data
Objectives	Any engineer or operator can access the historical process data.

Resources/Scope	Any process data to which the person should have access.				
	Explicit Access Attempt/Request info			Context Conditions	
#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny
1.	Any engineer and operator	READ	Process Historian	From the work computers of engineers and from plant computers.	Permit
Rule Combining Algorithm	Permit Overrides				

Policy Id & Name	P2. Access for modifications to set points for process variables				
Objectives	Process control engineers and chief plant operators can access and change/override the set points				
Resources/Scope	Manipulated process values to which the process control engineers and chief plant operators should have access.				
	Explicit Access Attempt/Request info			Context Conditions	
#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny
1.	ANY	READ	Process Historian	From the work computers of engineers.	Permit
2.	ANY	WRITE	Process Control tool	From the work computers of engineers.	DENY
3.	Process control engineers & chief plant operators	READ/ WRITE	Process Control tool	From the work computers of engineers and from plant computers.	Permit

Rule Combining Algorithm	Permit Overrides
--------------------------	------------------

Policy Id & Name	P3. Access for modifications to process alarm limits
Objectives	Process control engineers and chief plant operators can access and change/override/update the alarm limits of process variables
Resources/Scope	Alarm upper and lower limit values to which the process control engineers and chief plant operators should have access.

	Explicit Access Attempt/Request info			Context Conditions	
#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny
1.	ANY	READ	Process Historian	From the work computers of engineers.	Permit
2.	ANY	WRITE	Process Control tool	From the work computers of engineers.	DENY
3.	Process control engineers & chief plant operators	READ/ WRITE	Process Control tool	From the work computers of engineers and from plant computers.	Permit

Rule Combining Algorithm	Permit Overrides
--------------------------	------------------

Policy Id & Name	P4. Access to simple PID control logics
Objectives	Process engineers can access and define rules for process control
Resources/Scope	PID control logics which process engineers should have access.

Explicit Access Attempt/Request info			Context Conditions		
#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny
1.	ANY	READ	Process Historian	From the work computers of engineers.	Permit
2.	ANY	WRITE	Process Control tool	From the work computers of engineers.	DENY
3.	Process control engineers	READ/ WRITE	Process Control tool	From the work computers of engineers and from plant computers.	Permit
Rule Combining Algorithm		Permit Overrides			

Policy Id & Name	P5. Access to modifications to advance process control (model predictive control) logics				
Objectives	Process engineers can access and define rules for advance process control				
Resources/ Scope	MPC control logics which process engineers should have access.				
Explicit Access Attempt/Request info			Context Conditions		
#	Requestor	Action	Resource	<u>Contextual Attributes</u> Operator Parameters AND/OR	Permit / Deny
1.	ANY	READ/ WRITE	Process Control tool	From the work computers of engineers.	DENY
2.	Process control engineers	READ/ WRITE	Process Control tool	From the work computers of engineers and from plant computers.	Permit

Rule Combining Algorithm	Permit Overrides
--------------------------------	------------------

5. AquaSPICE Approach

5.1. RTM Architecture

The proposed architecture of the real time monitoring platform for AquaSPICE project has been presented in D3.1. As described, the proposed RTM platform is based on the open-source initiative FIWARE. This decision is because FIWARE is the only option addressing user requirements and needs. Specifically, FIWARE is the only system that exhibits compliance with NGSI-LD standards. Apart from the requirement fulfilment, FIWARE is considered to have a collaborative and mature ecosystem of developers (more than 8K), innovation Hubs (21), accelerators, cities, top members like Atos, Engineering, Red Hat, NEC, Telefónica and Trigyn Technologies among and more than 1000 SMEs and start-ups. Figure 2 presents the RTM Platform architecture.

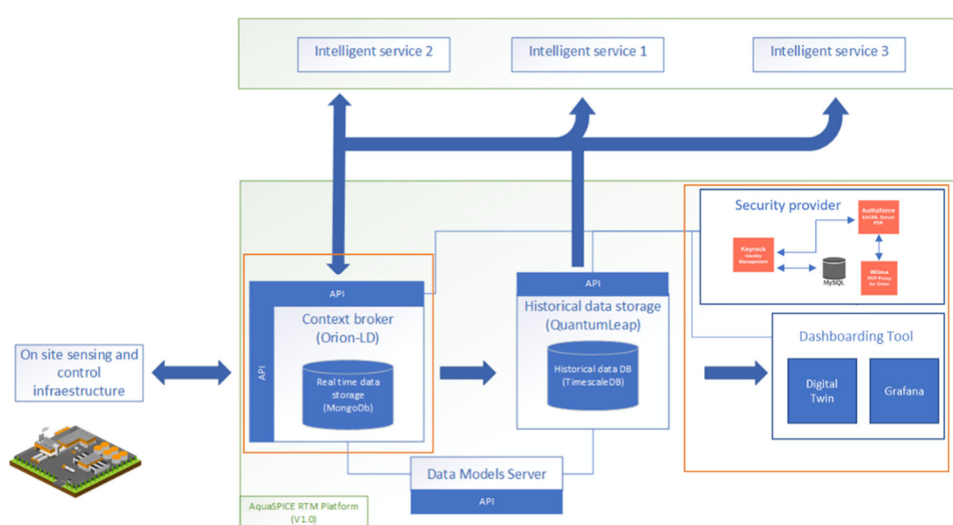


Figure 2: RTM Architecture

RTM Platform architecture diagram illustrates the fundamental structure of the platform and its integration within the AquaSPICE ecosystem. It consists of five components (i) Context broker, (ii) Historical data storage, (iii) security provider (iv) data model server and (v) Dashboarding tool.

5.2. Data Integrity

To guarantee that integrity of the data ingested by the RTM platform so that they can be confidently used by end users through dashboards as well as by intelligent services to reason over them, the context-broker described in D3.3 'Context-Broker Tool to Fuse and Share IoT' has in its core the Data Quality Assurance module (DataQA). This module evaluates the input data (coming from on-site sensors) in real time and produces transformed time series with quality metrics associated to the original readings (e.g., flag invalid readings). This evaluation is implemented for each data stream on each CS using a bag of algorithms and techniques offered by the DataQA module including feature engineering and outliers' detection and correction.

In WP3, we have been investigating, implementing and testing a variety of techniques for data integrity, including feature engineering such as examining extreme values, mean values, deviation values, higher moments of the statistical distribution, frequency domain analysis encompassing band amplitude and identification of maximum peaks. Outliers' detection and correction based on Zscore, InterQuartile Range (IQR), Hampel Filer and BDSCAN algorithms, and time series smoothing based on Moving Average, Exponential Moving Average and Gaussian Moving Average algorithms. Also, unsupervised machine learning algorithms such as Isolation Forest, Local Outlier Factor and Robust Covariance and anomaly detection pipelines based in Long Short-Term Memory (LSTM) recurrent neural networks have been investigated.

This ongoing work will continue as long as new sensors are deployed in the case studies and new data streams are transmitted to the RTM. Further tests and evaluations will provide evidence as to which algorithms are best suited to support Data Quality and Integrity with the AquaSPICE RTM and consequently WaterCPS systems.

5.3. Security Provider

This section is devoted at presenting the security provider service. The platform is responsible for enforcing access control for all client interactions, including device management, notification subscriptions, and historical data querying.

The access control mechanism empowers managers to establish a context-based set of rules, enabling relevant stakeholders to specify consent provisions and preferences to be applied either automatically or manually. The security provider is composed by three separated components:

- PEP-Proxy Wilma¹: Responsible for intercepting API requests and communicate with Keyrock for authentication and AuthZForce for authorization.
- Keyrock²: It is responsible for managing identities, enabling OAuth2-based authentication for both users and devices, handling user profile management, ensuring privacy-preserving handling of personal data, facilitating Single Sign-On (SSO), and supporting Identity Federation across multiple administration domains. Keyrock offers a range of features, which include the following:
 - It presents an OAuth2 authentication system that caters to both Applications and Users.
 - It provides a user-friendly interface designed specifically for managing identities.
 - Additionally, it incorporates a REST API that allows Identity Management operations to be performed through HTTP requests
- AuthZForce³: Responsible for enforcing security policies expressed using the XACML standard.

¹ <https://fiware-pez-proxy.readthedocs.io/en/latest/>

² <https://fiware-idm.readthedocs.io/en/latest/>

³ <https://authzforce-ce-fiware.readthedocs.io/en/latest/>

The decision of selecting the triad of components (Wilma, Keyrock and AuthZForce) is because these components are the de-facto standard for the development of authentication/authorization mechanisms in FIWARE-based architectures.

Once a request to any exposed REST service is intercepted by PEP-Proxy Wilma the authentication information provided by the client (user credentials, session token, etc..) will be extracted and diverted to Keyrock. Keyrock will validate the provided credentials and return the requester information. Then, a call from Wilma to AuthZForce will trigger the authorization of the request against security policies defined using XACML notation. If the process succeeds, the original request will be proxied to the destination API. If not, an authentication/authorization HTTP error will be returned to the originator of the request.

Authzforce is an advanced access control Generic Enabler within the FIWARE platform. It serves as an interpretive Policy Decision Point (PDP) that is capable of evaluating and enforcing access control rules defined using the XACML (eXtensible Access Control Markup Language) standard. With Authzforce, you can leverage its capabilities to interpret and apply access control policies specified in XACML, providing a powerful and flexible solution for managing access to resources within your FIWARE-based applications.

We successfully integrated Authzforce into a web application and effectively showcased examples of interactions between the Authzforce XACML Server and the Policy Decision Point (PDP). We also spined off a Grafana installation, and integrated it with Keyrock IDM. Users are able to login using their Keyrock credentials. This Grafana installation is available at the following link: <http://46.101.151.68/grafana>. The FIWARE app is deployed at the following link: <http://46.101.151.68:3000/>. The Keyrock IDM is at the following link: <http://46.101.151.68/idm/>. We expose Keyrock and Grafana through a reverse proxy (nginx).

The Grafana login page is presented in Figure 3.

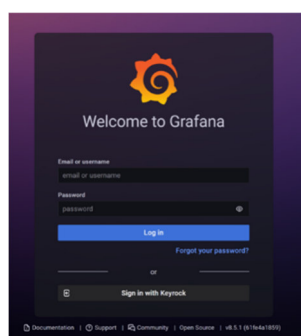


Figure 3: Grafana login page

The Grafana redirects users to the login page of the Keyrock Identity Manager in order to sign in. The Keyrock login page is presented in Figure 4.

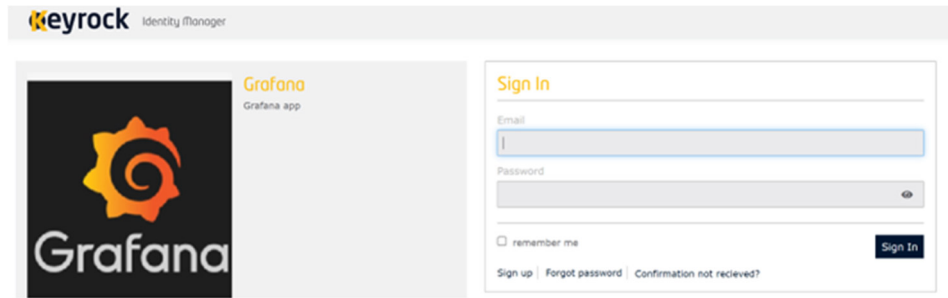


Figure 4: Login page of the Keyrock Identity Manager

5.4. Securing Access

To ensure secure access to application resources, two pieces of information are essential. Firstly, it is important to know the identity of the requester, and secondly, to determine if the requester has the permission to access the desired resource. The FIWARE Keyrock implements OAuth2, which is an industry-standard protocol for access delegation. It enables third-party applications to acquire restricted access to services. OAuth2 facilitates the resource owner, such as a plant engineer, to authorize a third-party application like Grafana Application to access specific information, such as device data, through a resource provider.

Authentication using a OAuth2 Challenge-Response mechanism is a requirement for both users and applications. After a successful authentication, the user is provided with a token that must be included in next requests. This token serves the purpose of identifying both the user and the associated application, while also specifying the user's access privileges. Keyrock, in conjunction with other enablers, can be used to restrict and control access.

The rationale behind OAuth2 is to avoid sharing one's own credentials with a third party for granting access. Instead, access can be granted at a specific level, either read-write or read-only. Additionally, it is possible to revoke access at any given time, granting the resource owner the ability to retain control over the individuals who have permission to access specific resources.

When the application successfully authenticates users, access can be further restricted through access control mechanisms. Access control requires the implementation of access policies, which define the permissions of different entities regarding specific actions.

5.4.1. Keyrock Identity Management

The Keyrock Identity Management database encompasses the following common objects:

User: Represents any registered user who can authenticate using his credentials. Users can be granted individual or group-based rights.

Application: Refers to a FIWARE application comprising a collection of services.

Organization : Represents a user group that can be allocated a specific set of permissions. Modifying the organization's rights affects the access of all its members.

OrganizationRole: Users can hold either the "member" or "admin" role within an organization. Admins have the authority to remove or add users from the organization, while members acquire the roles and permissions assigned to the organization. This structure enables effective management of its members, eliminating the necessity for a super-administrator to oversee all rights.

Role: Serves as a descriptive category for a set of permissions. Roles can be assigned to individual users or organizations. When a user signs in, they obtain permissions that are derived from both their individual roles and the roles linked to their organization.

Permission: Denotes the ability to perform certain actions on system resources.

Within a FIWARE application, it is also feasible to secure two additional application entities:

PEPProxy: Acts as middleware that validates user rights between generic enablers.

IoTAgent: Functions as an intermediary between IoT sensors and the Context Broker.

5.4.2. PDP Access control

PDP Access Control operates at three levels:

Authentication Access: Grants all privileges to signed-in users while restricting all actions for anonymous users.

Basic Authorization: It determines the specific resources and verbs that the logged-in user is authorized to access.

Advanced Authorization: Provides advanced and precise control using the eXtensible Access Control Markup Language (XACML). This level enables fine-grained control over access through the utilization of XACML.

Keyrock can independently offer a simplified Level 1 and 2 PDP. To achieve Level 3 access control, Keyrock can be combined with additional generic enablers. By integrating Authzforce into the existing security microservices (IDM and PEP Proxy) within the Smart Application infrastructure, Level 3 access control can be implemented effectively.

Authentication Access

Once a user successfully logs in with Keyrock or any other OAuth2 provider, they receive an `access_token` which confirms their existence. This `access_token` alone is enough to authenticate the user. To ensure the `access_token`'s freshness, a GET request can be made to the `/user` endpoint. If the response is successful, it indicates that the `access_token` is valid.

Basic Authentication

Authorization is the process of specifying access rights or privileges to ensure information security. In the context of FIWARE Keyrock, the Generic Enabler controls identity management, and grants user access based on assigned permissions tied to roles.

Keyrock allows each secured application to define a specific set of permissions, that represent actions within the application. For example, accessing a device may require a "Read data" permission, while interacting with the device may require a "Write data" permission. These permissions are grouped into roles, such as the "Plant Engineer" role, which may have both "Read data" and "Write data" permissions assigned to it.

Permissions can have overlapping characteristics and be assigned to multiple roles simultaneously. For instance, the "Read data" permission could also be assigned to the "Management" role. Users or organizations are then assigned to one or more roles, accumulating the permissions associated with each role they possess. Users are unaware of roles and only know the list of permissions granted to them.

After a user successfully signs in and acquires an `access_token`, it can be stored in a session for later use in retrieving user details. The request to retrieve user details can include resource permissions. Leveraging this information, access to individual resources can be granted or denied based on the user's permissions.

To summarize, permissions encompass all possible actions on application resources, while roles represent groups of actions that can be performed by specific user types. By utilizing the `access_token` and user details, access to resources can be controlled accordingly.

Advanced Authorization

To handle complex access control scenarios, an arbitration microservice is necessary. This microservice interprets and evaluates the full set of access control rules to make Permit/Deny policy decisions based on the evidence presented by the requesting service.

FIWARE Authzforce is an advanced access control Generic Enabler that serves as an interpretive Policy Decision Point (PDP). It can interpret rules defined using the XACML (eXtensible Access Control Markup Language) standard. The rulesets can be modified and uploaded at any time, providing a flexible approach to maintaining security policies that can adapt to changing business needs. The language used in describing the access policy is highly extensible and can accommodate various access control scenarios.

XACML policies are organized in a hierarchical structure consisting of three levels `<PolicySet>`, `<Policy>`, and `<Rule>`. At the highest level, a `<PolicySet>` encompasses multiple `<Policy>` elements, each of which can contain `<Rule>` elements.

To examine if access to a resource should be granted, the evaluation process involves assessing each `<Rule>` within a `<Policy>`. The overall result of a `<Policy>` is determined by the combined results of its `<Rule>` elements. When there are conflicting `<Policy>` results, combining algorithms define which `<Policy>` takes precedence.

A <Rule> element comprises a <Target> and a <Condition>. For example, a <Rule> may state that access is permitted when a POST request is made to the /get-data endpoint, provided that the subject:role attribute is provided and its value is "security-role-0000-0000-000000000000".

```
<Rule RuleId="get-data-0000-0000-000000000000" Effect="Permit">
  <Description>GET Data</Description>
  <Target>
    <AnyOf>
      <AllOf>
        <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">/get-data</AttributeValue>
          <AttributeDesignator Category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource" AttributeId="urn:thales:xacml:2.0:resource:sub-resource-id" DataValue="get-data" />
        </Match>
      </AllOf>
    </AnyOf>
  </Target>
  <Condition>
    <Apply FunctionId="urn:oasis:names:tc:xacml:3.0:function:any-of">
      <Function FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal" />
      <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">security-role-0000-0000-000000000000</AttributeValue>
      <AttributeDesignator Category="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject" AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role" DataValue="security-role-0000-0000-000000000000" />
    </Apply>
  </Condition>
</Rule>
```

Figure 5: Example of a simple access rule

While this may seem verbose for a simple Verb-Resource access rule, XACML allows for more complex comparisons. Conditions can be defined at the attribute level and combined to perform intricate calculations. For instance, conditions may involve checking the time of day or verifying that a URL contains a specific string.

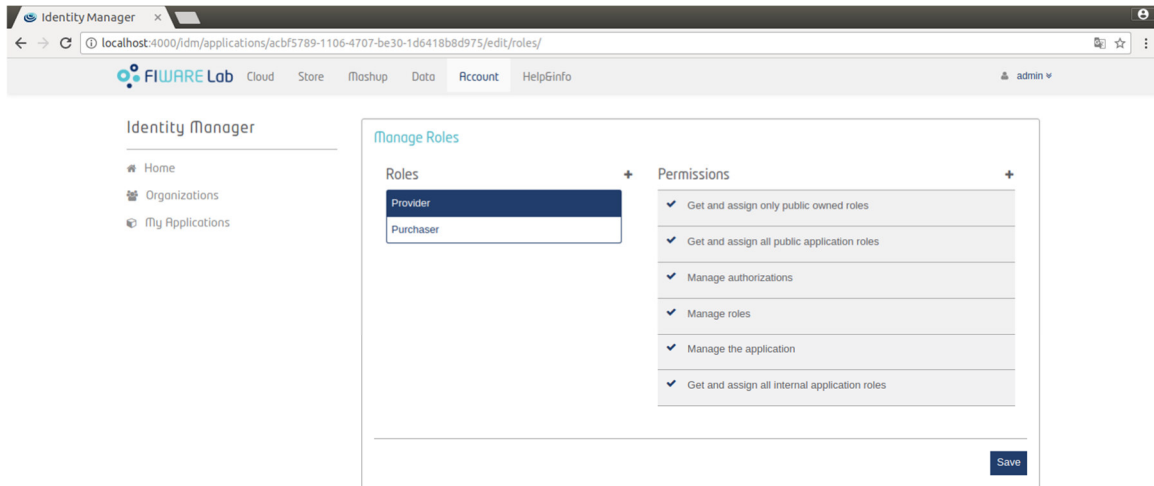
Advanced authorization facilitated by this approach allows for flexible rulesets. Permissions are no longer limited to fixed roles, resources, and actions. Instead, they can be expanded as needed. For instance, users that has a role X can access URLs starting with Z, provided the HTTP verb is GET, POST or PUT. Additionally, they can perform DELETE operations if they were the original creators of the resource.

In summary, by utilizing FIWARE Authzforce and the XACML standard, complex access control scenarios can be effectively managed, allowing for dynamic and customizable authorization policies.

5.5. Manage Roles Tutorial

In this section, we explain how to configure rules and define permissions using Keyrock.

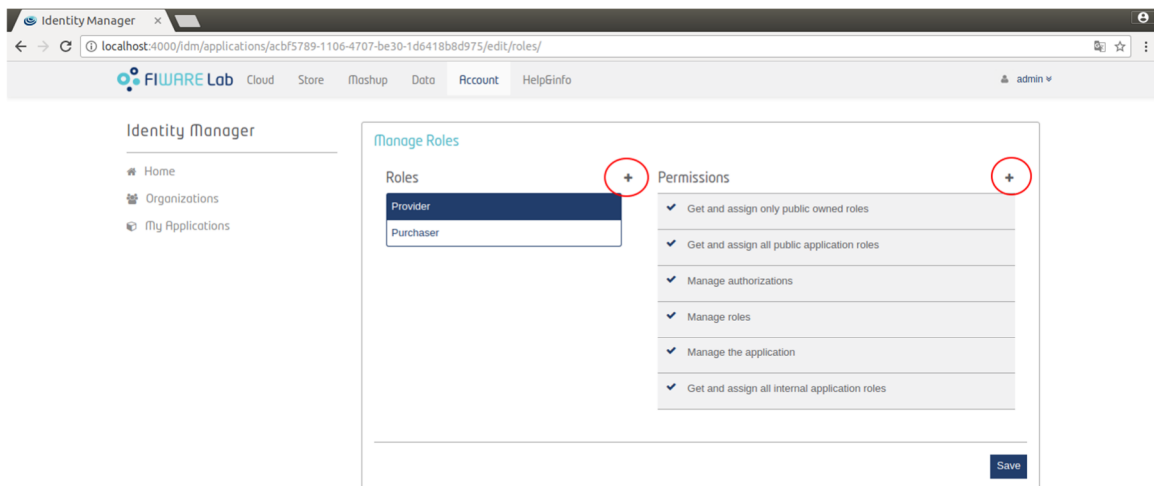
Keyrock offers two default roles: Provider and Purchaser. Clicking on either of these roles will display the permissions assigned to that particular role.



2015 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the [Terms and Conditions](#), [Privacy Policy](#) and [Cookies Policy](#).

Figure 6 Keyrock - roles to permissions mapping

Keyrock IDM offers the capability to create custom roles and permissions.



2015 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the [Terms and Conditions](#), [Privacy Policy](#) and [Cookies Policy](#).

Figure 7: Create roles and permissions

To create a new role in Keyrock, simply click on the "New role" option and provide a name for the role. Once you have entered the desired name, click on the "Save" button to create and save the new role. This process enables you to easily define and add custom roles within the Keyrock IDM interface.

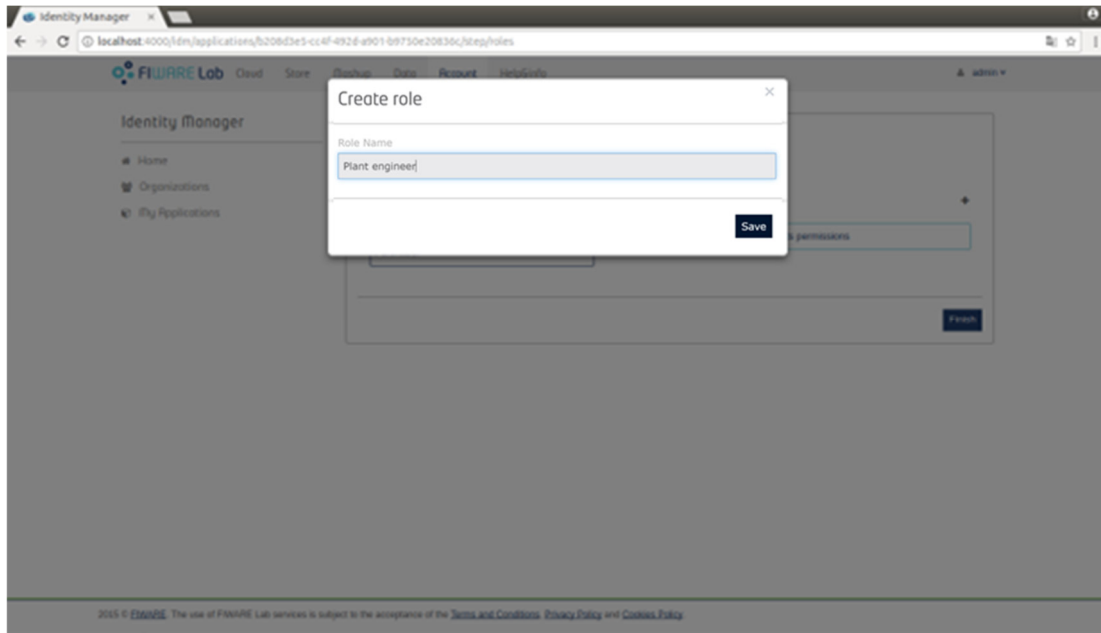


Figure 8: Create a role

In addition, you have the option to add new permissions within Keyrock by clicking on "New Permission." When creating a new permission, you will be prompted to provide the name of the permission, a description, the corresponding HTTP verb (such as GET, PUT, POST, DELETE), and the specific path associated with that permission. After entering these details, click on "Create Permission" and then "Finish" to complete the process of creating the application.

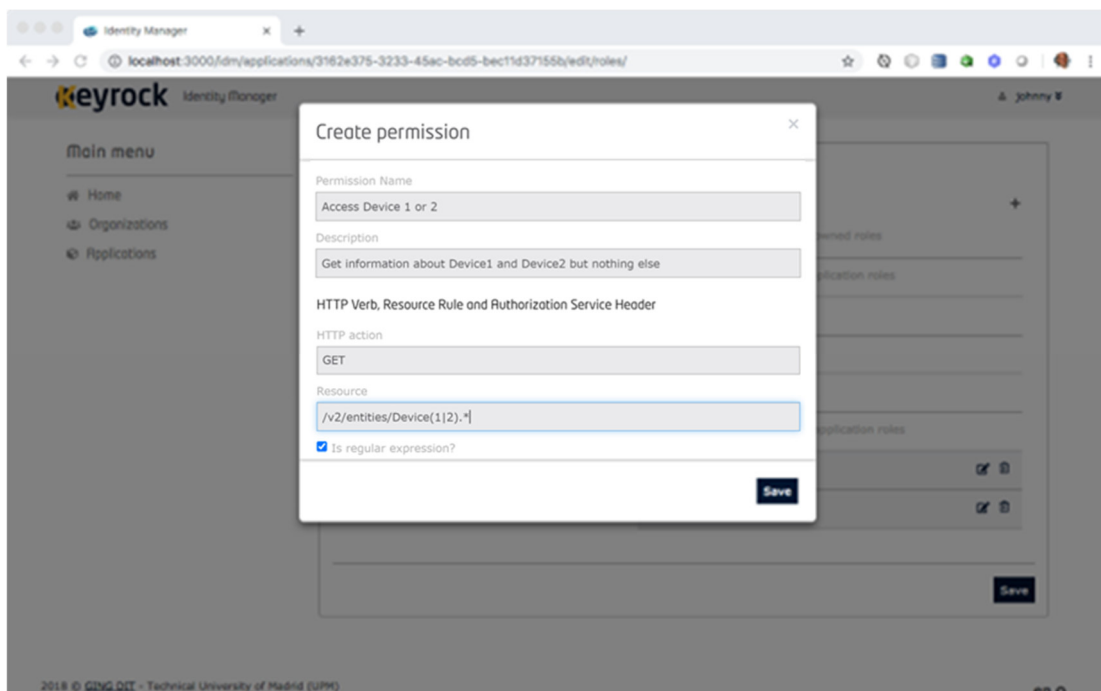


Figure 9: Create permission

If necessary, you can also configure a specific XACML rule within Keyrock.

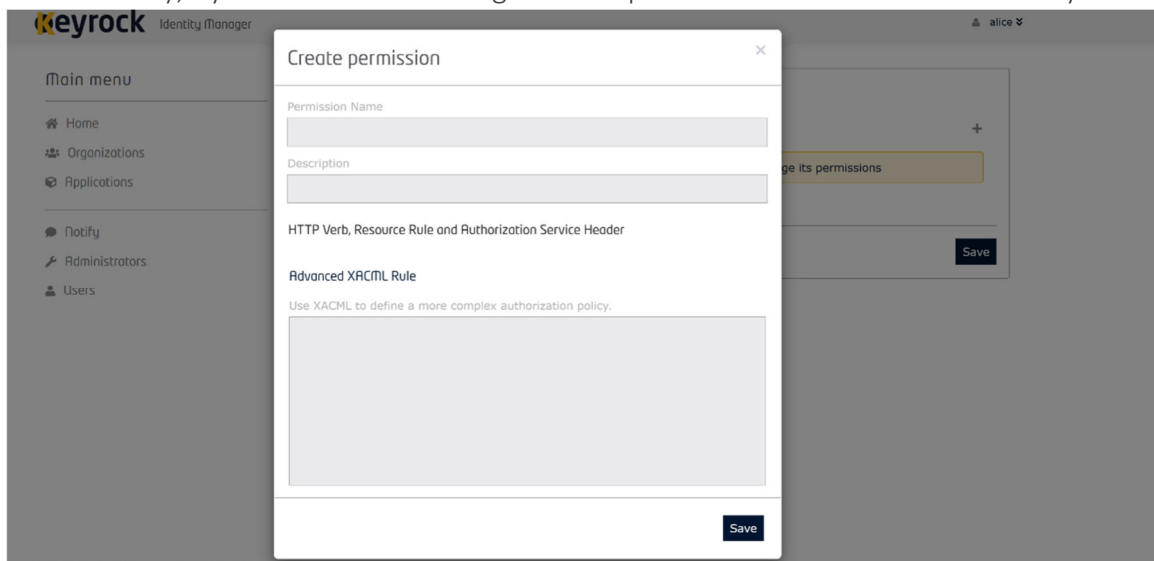


Figure 10: Create advanced XACML Rule

Furthermore, Keyrock provides the ability to edit and delete the roles and permissions you have created. You can easily access these options by clicking on the corresponding buttons within the Keyrock interface. To configure the permissions for a new role, you can activate the appropriate checkboxes associated with the desired permissions.

Once you have selected the necessary permissions, simply click on the "Save" button to create the new role assignment.

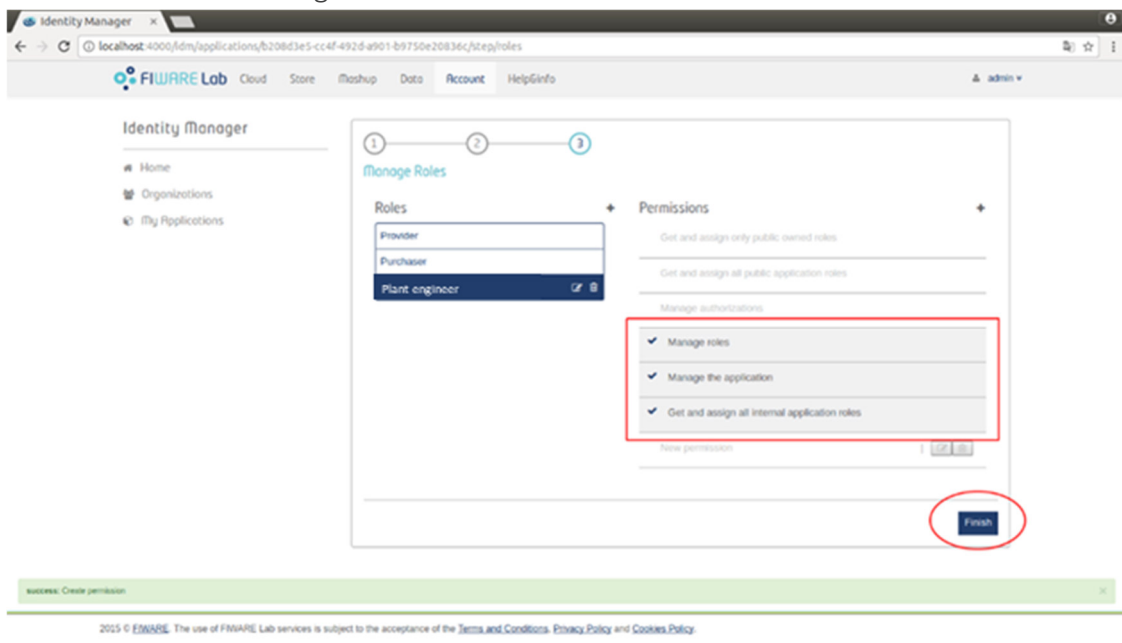


Figure 11: Keyrock - New role and assignments

5.6. Demo Example

This section presents a demo example that shows the interaction with smart devices in the context of municipal water utility.

The following users have accounts within the FIWARE web application:

User	Role	Username	Password
Alice	Administrator of the Keyrock Application	alice-the-admin@test.com	Test
Bob	Project Manager	bob-the-manager@test.com	Test
Charlie	IT Manager	charlie-security@test.com	Test
Rob	Data controller	rob@test.com	Test
Tom	Technician	tom@test.com	Test
Eve	No role	eve@test.com	Test

Authzforce comes preconfigured with a set of simple and fundamental role-based rules:

- Access by authorized personnel ONLY AND
 - Read/Write Access by the project manager, IT manager and Data controller ONLY
 - Read Access by Technician

Bob has the project manager role. He can access and interact with the devices of his assets.

Charlie has the IT manager role. He can access and interact with the devices of his assets.

Rob has the data controller role. He can access and interact with the devices of his assets.

Tom has the technician role. Access to retrieve device data is permitted, however access to interact with the device data is denied.

Eve has an active account in the application but does not have any assigned roles. While access to the application is permitted for all users who have successfully logged in, Eve is denied access to retrieve device data because she does not possess a role that grants permission for such access.

Instructions on how to install all the components presented in this section are described in the Appendix.

6. Conclusions and Next Steps

Based on the proposed architecture of the real time monitoring platform for AquaSPICE project exposed under D3.1 and security requirements emanating from the AquaSPICE Case Studies, this deliverable presents the development of a policy model that formally describes flexible context-based access control and consent rulesets. It also addresses static policies concerning the usage of AquaSPICE data generated by the RTM platform.

This deliverable reports on the state-of-the-art security approaches for context-aware access policies. It also investigates the security requirements to be considered in order to ensure the security of water data within the AquaSPICE RTM platform. The deliverable describes the proposed security provider service, explains how to configure rules and define permissions using the Keyrock system and presents a demo example that shows the interaction with smart devices.

Based on the approach described herein, Task 3.4 “Context-Aware Access Control, Data Integrity and Security Assurance Mechanisms” has implemented the security provider service that is responsible for enforcing security policies. The access control service ensures the enforcement of access control for all client interactions, including device management, historical data querying, and notification subscriptions. The service provides the capability to establish context-based rules, allowing relevant stakeholders to define consent provisions and preferences that can be applied automatically or manually.

7. Appendix: Security Questionnaires & Technical Instructions

7.1. Questionnaire for security requirements for context-aware access policies

Author:

<input type="checkbox"/>						
Policy Id & Name		E1.				
Objectives						
Resources/Scope						
Scenario						
Explicit Access Attempt/Request info				Context Conditions		
#	Requestor	Action	Resource	Contextual Attributes	Operator Parameters	AND/OR
1.						
2.						
3.						
Rule Combining Algorithm		Permit Overrides				

7.2. Instructions

To install docker, you should run the following commands:

```
sudo apt-get remove docker docker-engine docker.io containerd runc
```

```
sudo apt-get install \
```

```
ca-certificates \
```

```
curl \
```

```
gnupg \
```

```
lsb-release
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o  
/usr/share/keyrings/docker-archive-keyring.gpg
```

```
echo \
```

```
"deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-  
keyring.gpg] https://download.docker.com/linux/ubuntu \
```

```
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
sudo apt-get update
```

```
sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

The following command installs nginx:

```
sudo apt install -y nginx
```

After that you should add the section in nginx-config.txt using the following pico command

```
sudo pico /etc/nginx/nginx.conf
```

```
sudo systemctl reload nginx
```

To start the FIWARE installation, run the following:

```
git clone https://github.com/Fiware/tutorials.XACML-Access-Rules.git
```

After that, the services can be initialized by executing them from the command-line

```
cd tutorials.XACML-Access-Rules.git
```

```
./services create
```

```
./services start
```

After that you can create the Grafana app using Keyrock credentials. Note that Keyrock is by default in port 3005.

To install Grafana on your Ubuntu server, you have two options: downloading it directly from the official website or using an APT repository. In this tutorial, we will use the APT repository method as it simplifies the installation process and facilitates future updates.

Start by downloading the Grafana GPG key using the `wget` command. Then, pipe the output of the download command to `apt-key`.

```
wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
```

In the provided command, the `-q` option is used with `wget` to disable status update messages, while the `-O` option specifies that the downloaded file should be output directly to the terminal. The next step is to add the Grafana repository to your APT sources:

```
sudo add-apt-repository "deb https://packages.grafana.com/oss/deb stable main"
```

To ensure that you have the latest package information from the newly added Grafana repository, it is necessary to refresh your APT cache. This will update the package lists stored on your Ubuntu server. To refresh the APT cache, execute the following command:

```
sudo apt update
```

Now that you have added the Grafana repository and refreshed your APT cache, you can proceed with the installation of Grafana on your Ubuntu server. To install Grafana, run the following command:

```
sudo apt install grafana
```

More details for Grafana installation can be found in the following tutorial: <https://www.digitalocean.com/community/tutorials/how-to-install-and-secure-grafana-on-ubuntu-20-04>

After that you should configure OAuth2 authentication service with Grafana.

Run the following command:

```
sudo pico /etc/grafana/grafana.ini
```

And set according to <https://grafana.com/docs/grafana/latest/auth/generic-oauth/>

Also see <https://stackoverflow.com/questions/54998226/single-sign-on-keyrock-grafana-doesnt-work>.

Finally, follow <https://grafana.com/tutorials/run-grafana-behind-a-proxy/#configure-nginx> to deploy using nginx in `/grafana` sub-path, remote `http_port` from redirect url.

8. References

- [1] Seol K, Kim Y-G, Lee E, Seo Y-D, Baik D-K. Privacy-preserving attribute-based access control model for XML-based electronic health record system. *IEEE Access*. 2018;6:9114–28.
- [2] Oasis-open.org. [cited 2021 Sep 16]. Available from: <http://docs.oasis-open.org/xacml>
- [3] Quiroigico S, Hu V, Karygiannis T. Access control for sar systems. 2011.
- [4] Open Policy Agent [Internet]. Openpolicyagent.org. [cited 2021 Sep 27]. Available from: <https://www.openpolicyagent.org/>
- [5] Siebach JAJ. The Abacus: A New Approach to Authorization. Brigham Young University; 2021.
- [6] Verginadis Y, Patiniotakis I, Gouvas P, Mantzouratos S, Veloudis S, Schork ST, et al. Context-aware policy enforcement for PaaS-enabled access control. *IEEE trans cloud comput*. 2019;1–1.
- [7] Dennis JB, Van Horn EC. Programming semantics for multiprogrammed computations. *Commun ACM*. 1966;9(3):143–55.
- [8] Gusmeroli S, Piccione S, Rotondi D. A capability-based security approach to manage access control in the Internet of Things. *Mathematical and Computer Modelling*. 2013;58(5-6):1189-1205.
- [9] Gong L. A secure identity-based capability system. In: *Proceedings 1989 IEEE Symposium on Security and Privacy*. IEEE Comput. Soc. Press; 2003.