An Integrated Access Control Service Enabler for Cloud Applications

Tran Quang Thanh[1], Stefan Covaci[1], Benjamin Ertl[1] and Paolo Zampognano[2]

[1] Technical University Berlin, Germany
{tran, stefan.covaci, benjamin.ertl}@tu-berlin.de
[2] Engineering Ingegneria Informatica S.p.a., Italy
paolo.zampognaro@eng.it

Abstract. Cost reducing, ubiquitous access, are foreseeable benefits when organizations outsourcing applications, services to the cloud. However, security is a current major challenge that limits their widespread deployments. In this paper, a RESTful security service enabler is proposed to provide authentication, authorization, and audit logging services for cloud application developers, by leveraging several important security standards (e.g. OAuth, XACML). Specifically, a prototype of this enabler is ongoing developed based on our requirement investigation in the health care domain and related Generic Enabler technologies in the FI-PPP (Future Internet Public Private Partnership) FIWARE Project.

Keywords: Access Control, Cloud, eHealth, Future Internet, Security.

1 Introduction

Migrating applications to the cloud is getting a lot of interests in many organizations by its given benefits regarding flexibility, scalability and cost-effective. Cloud technology also changes the way of how applications and services are developed: from software-enabled to API-enabled. Considering as the evolution of SOA (Software Oriented Architecture), API technology enables organizations to better support their developers and to introduce their applications/services to a wider range of customers and partners. As critical applications and sensitive data are moving to the cloud, accessible through given APIs, it becomes clear that proper security solutions should be in place to protect them from malicious usages.

In this paper, an integrated access control service enabler is introduced to help cloud application developers when implementing and deploying their services. By consuming exposed APIs, developers save a lot of time when developing security logics as required credentials, access policies and decision support engines are managed outside the applications. In our solution, all security functionalities including identity provisioning, authentication, authorization and audit logging are provided as REST APIs [1]. In addition to that, several important security standards are adopted (e.g. SAML [2], XACML [3], OAuth [4], SCIM [5] and OpenID Connect

[6]). It is worth to mention that our proposed enabler is flexible enough, not only on how it will be implemented (as different existing software components can be reused) but also on the capability to deploy in different service domains. As an illustration, a prototype of this enabler has been ongoing developed and evaluated in the context of FI-STAR project [7].

This paper is organized as follows. Section II gives an overview of relevant standards and works. The architecture, APIs, and service functionalities are presented in Section III. A prototype of this enabler is discussed in Section IV based on our requirement investigation in the health care domain and related Generic Enabler technologies in the FI-PPP (Future Internet Public Private Partnership) FIWARE Project [8].

2 Background

OAUTH [4] (Open Authorization framework) is the evolving standard to secure API access. OAuth allows users (resource owner) to grant third-party applications (client) accessing data (resource server) without sharing credential (e.g. password). The client can be a native mobile application, a desktop application, a server-side or browser-based web application. The general flow of current OAuth version 2.0 is shown in Figure 1. The client interacts with resource owner to get authorization grant and use this to authenticate and get an access token from authorization server (which owns the user identities and credentials). Such token is then used to request resource access. The access token has often limited rights and can be revoked any time.

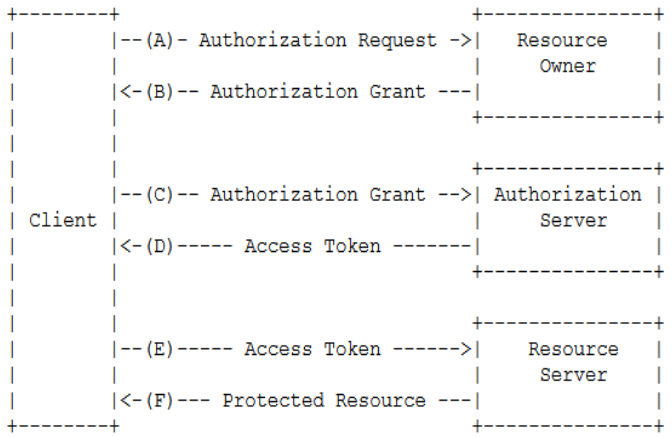

Fig. 1. OAuth2 abstract protocol flow

OAuth has widespread adoption by many service providers (e.g. Google, Twitter, FaceBook, GitHub, etc.). The current version 2 (OAuth2) keeps the overall architecture but is not compatible with previous versions (OAuth 1.0 and OAuth 1.0a). **OpenID Connect** [6] defines a simple identity layer on top of the OAuth2. It allows Clients to verify the identity of the End-User based on the authentication

performed by an Authorization Server, as well as to obtain basic profile information about the End-User in an interoperable and REST-like manner. The current version of Open ID Connect is 1.0, and it is not compatible with previous OpenID versions.

SCIM [5] System for Cross-domain Identity Management (former as Simple Cloud Identity Management) defines a RESTful protocol for identity management operations. It consists of three main elements: SCIM Core schema, REST API (for exchange user-related resources via XML or JSON) and binding (e.g. SAML binding). OAUTH2 is recommended for SCIM API authentication and authorization. SCIM has widely supported by Google, Ping, and Salesforge etc.

SAML [2] The Security Assertion Mark-up Language (SAML) is an XML-based framework that allows identity and security information to be shared across security domains. SAML is also the industry standard for the SSO mechanism for cloud and/or enterprise applications. SAML 1.0 and 1.1 has specified by OASIS (Organization for the Advancement of Structured Information Standards) but the today's SAML 2.0 is the convergence of three standards: SAML 1.1, Liberty Alliance Identity Federation Framework (ID-FF) and Shibboleth.

XACML [3] eXtensible Access Control Markup Language (XACML) is developed by OASIS to standardize the authorization decisions in enterprise applications. XACML defines a XML-based policy language, a request / response scheme, and access control architecture. The decision-based architecture is presented in figure 2 with different components such as PEP (Policy Enforcement Point) or PDP (Policy Decision Point). Many organizations are using XACML, and the latest version is 3.0

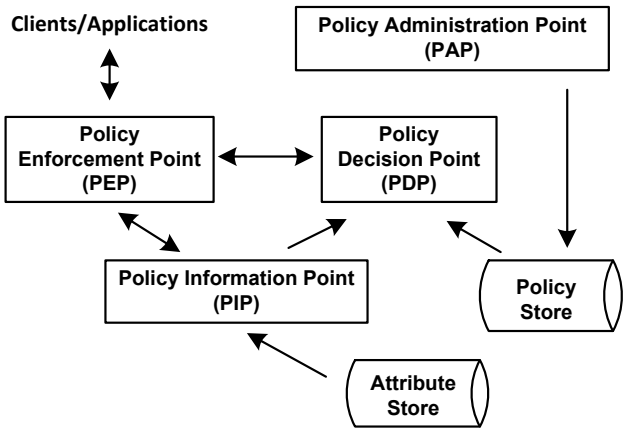

Fig. 2. XACML reference architecture

SYSLOG [9] is a de factor message logging standard for many years. It is widely adopted for logging network and security event by the capability to separate software that generates messages, storage systems, and reporting and analyzing solutions. The latest technical specification was published in 2009 by IETF with some significant enhancements.

Related Works:

The concept of "Enabler" came from the Future Public Private Partnership program (FI-PPP) [10] by European Commission to accelerate the development and adoption of Future Internet technologies. As specified, an enabler is a technological component that provides set of APIs and interoperable interfaces to support a concrete set of functions. There are many research projects and partners involved in the program. In the FI-PPP, FIWARE [8] is the core project that specified and implemented a lot of general-purpose enablers (GE) that common to almost usage areas. Such GEs are categorized into primary technical chapters: Cloud Hosting, Data & Context Management, Internet of Thing (IoT), Application, Advanced Middleware and Interface to Network and Device and Security. FI-PPP has also supported some use case projects in several important domains such as SAFECITY (smart city), FINESCE (smart energy), FINEST (transport) and FITMAN (manufacturing) [10]. In these projects, besides leveraging the usage of existing FIWARE GEs, they have also introduced new specific enablers (SE). About two hundred GEs and SEs have been specified, and the number will continuously increase as the program is still evolving. The enabler in this work is designed based on a set of software components (enablers) including the two related FIWARE GEs: Identity Management GE (IDM) [11] and Authorization PDP GE [12]. The integrating approach provides several advantages, not only better functionality support but also more flexible and convenience. The detail is further discussed in the next section.

3 Architecture

Developers prefer to spend time on business logic rather than security aspect. Utilizing existing security supported modules, software components is often the choice, especially for non-security developers. For their access control requirement, they either embed the authorization logic within the code (e.g. using an authentication/authorization security framework) or outsource to another external component (e.g. Identity and Access Management - IAM or IDM). By adopting the latter approach, not only the ease-of-implement and time-saving target are achieved, but also making applications easier to maintain, evolve and more flexible to adapt to new requirements especially when they will be deployed in the cloud environment. In this section, an "Integrated Access Control Enabler" (IAC) is described which takes into account most-wanted security requirements from cloud application developers by combining the Authentication, Authorization, and Auditing (AAA) services and leveraging several selected enterprise and cloud security standards. AAA are often the required functionalities in any application for security reason. Figure 3 gives an overview of the high-level architecture of the IAC. In order to keep our enabler cloud-ready, flexible and easy-of-implement, following design principles are adopted:

- *API-centric*: Service functionalities are exposed through a robust set of REST APIs. REST architecture style is widely adopted by its given benefits to application/service developers such as simple (e.g. using HTTP GET, POST, PUT and DELETE for Create, Retrieve, Update and Delete

operations), standard-aligned (using standards such as HTTP, URI, XML and JSON) and protocol independent (not only coupled to HTTP). As shown in Figure 3, different groups of APIs are provided including Authentication APIs (identifying and authenticating), Authorization APIs (deciding whether resource access is allowed), Audit Logging APIs (collecting access log for subsequent review or malicious usage detection) and Provisioning APIs (managing identity, access control policy).

- *Standard-aligned*: Selected security standards are investigated and taken into account: Authentication (SAML, OAuth2, and OpenID Connect), Authorization (OAuth2, XACML), Audit Log (Syslog) and Identity Provisioning (SCIM).
- *Modular & Layered*: As shown in the architecture, the IAC enabler can be built based on five main functional blocks (enablers). The interactions between them are through APIs or standard protocols. Detail information of such enablers are described as follows:

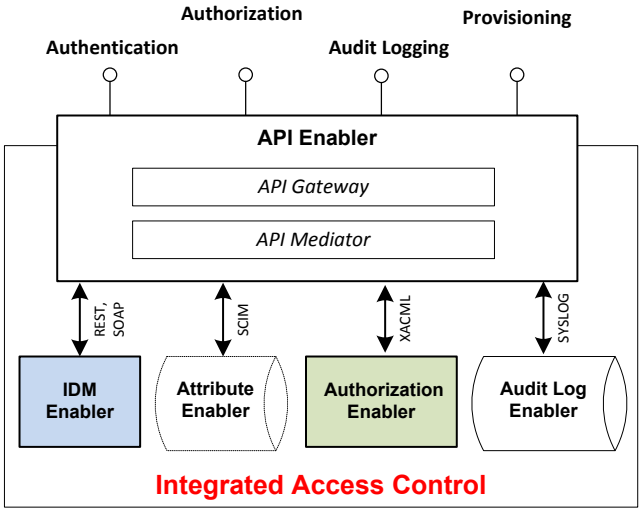

Fig. 3. High-level architecture of IAC Enabler

The front-end **API Enabler** controls which APIs are available and how to access them. It consists of two modules: *API Gateway* and *API Mediator*. The former protects exposed APIs from unauthorized attacks and misuses through different mechanisms such as input validation, rate limiting, and token-based access control. The latter interacts with back-end enablers to handle the API request. Depending on the back-end capability, there are two types of mediator behaviors: Redirect and Translator. The redirect action is applied when an API request can be served by a back-end enabler. In this circumstance, the API mediator replies the API request (or route it) with guiding information about new service end-point. For example, when the back-end "IdM Enabler" supports OAuth, all the OAuth requests to the front-end will be redirected to it. With other APIs that can not directly serve by any back-end

enabler, the translation is required as the front-end needs to handle API requests by interacting with back-end enablers through their given specific APIs. Such translation can be simple (e.g. REST-to-SOAP) or more complex (e.g. requiring multiple APIs correlations).

There are four main service enablers in the back-end: (OAuth-supported) IDM Enabler, (XACML) Authorization Enabler, (SCIM) Attribute Enabler and (Syslog) Audit Log Enabler.

- **IDM Enabler**: any OAuth-supported Identity Management System. The IDM Enabler should implement standard OAuth 2.0 grant flows: authorization code grant, resource owner password credentials grant, implicit grant, and client credentials grant. It is recommended to comply with existing security standards (section 2) and to provide administration tool for identity life-cycle management. Many existing IdM solutions can be used as IDM Enabler including any implementation of FIWARE IdM GE [11] or other open source solutions (e.g. WSO2 IS [13]).
- **Authorization Enabler**: any XACML authorization engine. This enabler provides the functionality of several components in the XACML access control architecture (figure 2) such as the policy decision point (PDP), policy administration point (PAP) and policy store (PRP). The PDP gives authorization decision based on various attributes and access policies. Through the PAP interface, administrators can manage access policies those are stored at PRP. Like the IDM, beside commercial solutions (e.g. Axiomatics Policy Server [14]), some open source candidates are also available such as SunXACML [15], FIWARE Authorization PDP [12] or WSO2 IS [13].
- **Attribute Enabler**: a secondary user attribute management and storage. It is an optional component as its functionality can be part of the IDM enabler. However, as many IDMs support only a fix set of common user attributes, the requirement to support other specific attributes is fulfilled by using this enabler. Supporting different types of attributes is important for any access control solution to be flexible enough to adapt to requirements from various service domains. LDAP or RDBMS can be used for data storage, and management APIs are recommended to comply with SCIM standard [5].
- **Audit Log Enabler**: any Syslog server and storage. However, this enabler leverages Syslog standard to collect only sensitive access events (e.g. authentication, authorization activities). Such logging information will help to maintain permanent evidence of all authorized and unauthorized access to protected resources.

Table 1 summarizes service interfaces that should be supported by any implementation of this enabler.

Table 1. Service Interfaces

Interface	Description

iOAuth	This interface supports four main OAuth2 flows as described in the OAuth2 specification RFC 6749 (authorization code grant, resource owner password credentials grant, implicit grant, and client credentials grant) This interface supports SAML Assertion OAuth flow, an ongoing extension of OAuth2 by IETF to support identity federation. Legacy systems often adopt SAML token for authentication Providing CRUD APIs for managing OAuth applications (CRUD: Create, Retrieve, Update, Delete) Providing APIs for access token validation
iOpenIDConnect	OpenID Connect authentication interface. OpenID Connect defines a simple identity layer on top of the OAuth 2.0. Supporting this interface is optional.
iSCIM	Providing SCIM APIs (System for Cross-Domain Identity Management Protocol) to manage identity.
iPDP	Providing REST APIs to evaluate XACML authorization request (Policy Decision Point interface)
iPAP	Providing REST APIs to manage XACML access control policies. (Policy Administration Point interface)
iAuditLog	Providing REST APIs for managing authorized and unauthorized access to protect resources.

In the IAC, the token-based access control model is adopted to provide access control service. To access protected resources, cloud applications/services must present an access token (OAuth2 token). Our enabler offers two options to check whether or not to grant the resource access:

- *Coarse-grained access control*: Only the OAuth access token is checked, and if it is valid (e.g. belonging to an authenticated user), the authorized decision is granted.
- *Fine-grained access control*: Both the token and other attributes of the token owners are checked. It is achieved by adopting XACML Attribute-Based Access Control (ABAC) and Policy Based Access Control (PBAC) model.

Figure 4 presents a typical scenario that using the access control service: A user uses a client (web, desktop, mobile application) to access cloud resource. First, he needs to authenticate with the Access Control Service to get an access token and then includes such token in the resource request. A specific module, PEP (Policy Enforcement Point), which can be part of the resource or located another intermediate security component (e.g. security gateway/proxy), will intercept the resource request and check with the access control service in order to grant or deny the access by applying the coarse-grained or fine-grained approach. Flexibility, Reusability are several foreseeable benefits when using such token-based access control method as access policies are specified and validated separately from where they are enforced.

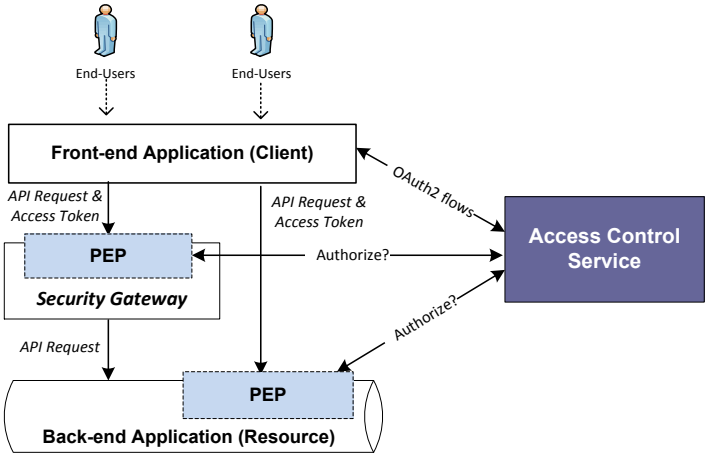

Fig. 4. Token-based Access Control Model

In addition to that, several emerging cloud-centric solutions are also integrated into our enabler including SCIM-based Identity Provisioning and OAuth-based Federated Authentication.

SCIM-based Identity Provisioning:

When outsourcing applications to the cloud, enterprise, and organization often want a simple solution that can leverage existing identities (e.g. from their legacy IDM systems) to give them access to cloud applications. Such credential (with all required attributes) must be provisioned/synchronized to the cloud using manual, just-in-time or proprietary approach. SCIM, the current evolving provisioning protocol, is adopted as it fulfills such requirement in a standard way and using REST. Moreover, beside standard attributes in the core schema, SCIM allow supporting new attributes by specifying them in an extension schema.


```
{
"userName":"test",
"password":"fistar",
"name":{"familyName":"Tran","givenName":"Quang Thanh"},
"addresses":{"country":"Germany","streetAddress":"Kaiserin Augusta Allee 31","postalCode":"10589"},
"phoneNumbers":[{"value":"4321","type":"mobile"},{"value":"1234","type":"other"}],
"ims":[{"value":"tran@skype.com","type":"skype"}],
"emails": [{"value":"tran@mail.tu-berlin.de","type":"work"}],
"profileUrl":"av.tu-berlin.de/thanh",
"preferredLanguage":["deutsch","english"],
"fistarExtension":
  {
  "emergency":"inactive",
  "relatedPerson": ["http://fistar.test/Users/person1","http://fistar.test/Users/person2"],
  "gender":"male",
  "dateOfBirth":"1981-01-01",
  "careProvider":"Hospital A",
  "device":["device1","device2"]
  }
}
```


Fig. 5. Example of SCIM core and specific attributes

Feature 5 describes how we extend the SCIM to support required specific attributes for our emergency use cases in the healthcare domain. Attributes inside green rectangle are the core SCIM attributes and those inside the red one are specific.

OAUTH-based Federated Authentication:
As an open authorization framework, OAuth also allows client application requesting the access token using an existing proof of authentication (federated authentication). Such feature is important as it can offer access control service to utilize federated identities (managed by another trusted identity provider). In our enabler, the SAML 2.0 Bearer Assertion Profiles for OAuth 2.0 specification [16] is taken into account as many existing identity providers using SAML assertion as user claim. Figure 6 explains how it works.

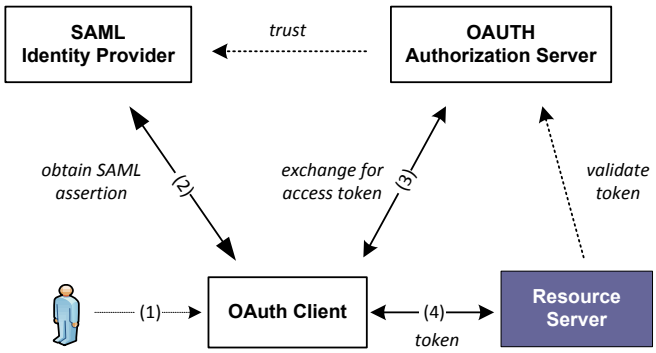

Fig. 6. SAML Grant Type for OAuth

(1) User requests access to protected resource from OAuth client (e.g. web, desktop or mobile application)
(2) User authenticates and obtains SAML assertion from a local SAML Identity Provider
(3) Client requests to OAuth AS to exchange the SAML assertion for an OAuth access token by using the SAML Grant type for OAuth grant flow. The OAuth AS needs to trust the SAML Identity Provider
(4) Received token is added to the API call to the (SaaS) resource.

4 Implementation and Result

In this section, an implementation of the proposed access control enabler is presented and currently under evaluation in the context of FI-STAR project [7]. Healthcare sector has been underway the major transition from specialist centered hospital to distributed patient-centric. More patients will be treated over the internet in their home, outside the hospital. The transformation, in one hand, is essential to make patient treatment more efficient, on the other hand, to help to reduce the cost.

Leveraging innovation technologies such as cloud computing will offer the ubiquitous data access and significant economic benefits.

The IAC enabler aligns well with the current healthcare standards as they are also interested in our selected security standards (e.g. OAuth2, SAML, Syslog etc.). We have investigated several related works such as IHE profiles [17] or HL7 FHIR specifications (Health Level 7 - Fast Healthcare Interoperability Resources) [18]. The former are specified by IHE (Integrating the Healthcare Enterprise), an initiative by healthcare professionals and industry, to improve the way computer systems in healthcare share information. In the latter, health care domain adopts REST for their resources specification. Through several human-related resources, various specific attributes can be selected to incorporate into our enabler implementation in order to adapt to requirements from healthcare organizations. Table 2 gives a first look at the potential mapping between our current supported SCIM user attributes with FHIR Patient Resource.

Table 2. Common SCIM and FHIR Patient Attributes

SCIM	FHIR Patient Resource
Core schema:	
id, username	identifier
name	name
phoneNumbers	telecom
addresses	address
preferredLanguage	communication
photo	photo
active	active
FI-STAR Extension Schema:	
gender	gender
birthDate	birthDate
careProvider	careProvider
…	…

A prototype of the IAC enabler is ongoing developed in the context of FI-STAR project [7]. The current beta version is published at [19]. In this release, the API Enabler is implemented using Java programming language and Spring framework [20]. For the back-end enablers, several open source software components are selected such as WSO2 Identity Server [13] and RSYSLOG [21]. It is worth to mention that the current implementation is flexible as all interactions among our components are through web APIs and standard protocols. It can be deployed in the cloud, on customer premise or even on the user computer. Figure 7 describes a SCIM API that allows getting user profile by given SCIM ID. We can see a lot of supported user attributes in the response message. Further information about supported APIs (e.g. user guide document, sample code) as well as upcoming updates will be continuously updated at [19].

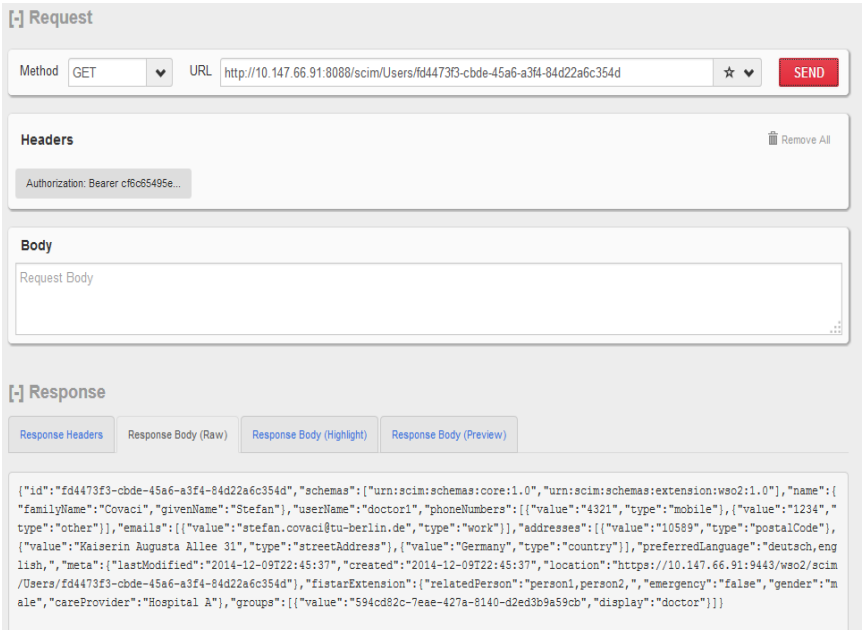

Fig. 7. Testing a SCIM API

5 Conclusions

Implementing security logic is always the complicated part in any application. In this paper, an integrated access control service enabler is proposed to help cloud application developers mitigating such challenge. Moreover, several widespread security standards are taken into account to fulfill emerging security requirements including identity provisioning, federated authentication, and flexible access control. In addition to that, by applying the standard and modularity design principle, the proposed enabler is flexible enough, not only on how it will be implemented (as its building blocks can be made use of existing software components), but also on the capability to deploy in different service domains. Specifically, a proof-of-concept prototype has been developed based on the requirements from healthcare domain in the context of FI-STAR project. For the future work, selected privacy enhancing technologies such as privacy-preserving authentication, pseudonymization will be investigated and developed.

Acknowledgment

The research leading to these results has received funding from the European Research Projects: FP7 FI-STAR and H2020 ARCADIA.

References

1. REST API Tutorial, http://www.restapitutorial.com
2. SAML Specifications http://saml.xml.org/saml-specifications
3. OASIS eXtensible Access Control Markup Language, https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml
4. OAuth 2.0 Authorization Framework, http://tools.ietf.org/html/rfc6749
5. System for Cross-domain Identity Management (SCIM), http://www.simplecloud.info
6. Open ID Connect, http://openid.net/connect
7. FI-STAR: Future Internet Social and Technological Alignment Research, https://www.fi-star.eu
8. FIWARE: Open APIs for Open Minds, http://www.fiware.org
9. The Syslog Protocol, https://tools.ietf.org/html/rfc5424
10. Internet-enabler Innovation in Europe, http://www.fi-ppp.eu/projects/
11. FIWARE Identity Management Open Specification, https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/FIWARE.OpenSpecification.Security.IdentityManagement
12. FIWARE Authorization PDP Specification, http://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/Access_Control_GE.Authorization.Open_RESTful_API_Specification
13. WSO2 Identity Server, http://wso2.com/products/identity-server/
14. Axiomatics Policy Server, http://www.axiomatics.com/solutions/products/authorization-for-applications/axiomatics-policy-server.html
15. Sun XACML implementation, http://sunxacml.sourceforge.net/index.html
16. SAML 2.0 Profile for OAuth 2.0 Client Authentication and Authorization Grants, http://tools.ietf.org/html/draft-ietf-oauth-saml2-bearer-23
17. Internet User Authorization Profile (IUA), http://wiki.ihe.net/index.php?title=Internet_User_Authorization
18. Fast Healthcare Interoperability Resources, http://www.hl7.org/implement/standards/fhir/resourcelist.html
19. Security & Privacy IAC (Integrated Access Control), http://catalogue.fi-star.eu/enablers/securityprivacy-iac
20. Spring Boot framework, http://projects.spring.io/spring-boot/
21. RSYSLOG: The Rocket-fast System for Log Processing, http://www.rsyslog.com