



From Real-world Identities to Privacy-preserving and Attribute-based  
CREDentials for Device-centric Access Control



WP5 – Attribute-Based Access Control

Deliverable D5.3 “Advanced Extensions: cryptographic attribute  
management, learning algorithms for complex ABAC reasoning, and privacy awareness tool”

**Editor(s):** CUT

**Author(s):** Savvas Zannettou, Harris Partaourides, Michael Sirivianos, Antonis Papasavva (CUT), Alberto Caponi, Claudio Pisa, Tooska Dargahi, Giuseppe Bianchi (CNIT), Vangelis Bagiatis, Kostas Flokos (UPCOM), Steven Gevers (VERI), Christoforos Ntantogian, Eleni Veroni, George Karopoulos, Nikolaos Koutroubouchos, Christos Lyvas, Panagiotis Nikitopoulos (UPRC), Alberto Carp, George Gugulea (CSGN), Vasillis Sarris (EXUS)

**Dissemination Level:** Public

**Nature:** Report



**Version:** 1.2

#### ReCRED Project Profile

Contract Number	653417
Acronym	ReCRED
Title	From Real-world Identities to Privacy-preserving and Attribute-based CREDentials for Device-centric Access Control
Start Date	May 1 <sup>st</sup> , 2015
Duration	36 Months

#### Partners

 University of Piraeus	University of Piraeus research center	Greece
 Telefónica Investigación y Desarrollo	Telefonica Investigacion Y Desarrollo Sa	Spain
	Verizon Nederland B.V.	The Netherlands
	Certsign SA	Romania
	Wedia Limited	Greece
	EXUS Software Ltd	U.K.
 Blinging business and IT together	Upcom Bvba (sme)	Belgium
	De Productizers B.V.	The Netherlands
	Cyprus University of Technology	Cyprus
	Universidad Carlos III de Madrid	Spain

	Consorzio Nazionale Interuniversitario per le Telecomunicazioni	Italy
	Studio Professionale Associato a Baker & McKenzie	Italy

### Document History

Version	Date	Author	Remarks
0.1	10/07/2017	CUT	TOC proposal
0.2	08/08/2017	CNIT, CSGN, EXUS, UPCOM, VERI CUT, UPRC, PROD	First round of contributions
0.3	12/08/2017	CUT	Integration of contributions
0.4	15/09/2017	CNIT, CSGN, EXUS, UPCOM, VERI CUT, UPRC, PROD	Second round of contributions and edits in contributions
0.5	17/09/2017	CUT	Integration of edits
0.6	25/09/2017	CUT	Addition of authors
0.7	25/09/2017	CUT	Review of document
1.0	25/09/2017	CUT	Final version
1.1	28/09/2017	CUT	Minor corrections
1.2	30/04/2018	CUT	Revised access control reasoning tool and revised consent management

## Executive Summary

This document is part of the WP5 (Attribute-Based Access Control) of the ReCRED project. The purpose of this deliverable is to report the definition and the description of advanced cryptographic extensions, the reasoning tools for complex ABAC policies as well as the risk awareness tools that are used in the ReCRED project. The deliverable complements and completes the previous WP5 deliverables. Specifically, D5.1 (Specification and Initial Design of the ABAC infrastructure) and D5.2 (Full design and prototype of the ABAC infrastructure).

The main purpose of this document is to provide a detailed description of the design and the implementation of components that provide to the ReCRED platform advanced cryptographic capabilities. First, we provide adequate information regarding the implemented advanced cryptographic extensions based on privacy-preserving access control primitives. Subsequently, we demonstrate the implementation of some advanced tools that assist the users in using and better understanding complex policies and risks. Moreover, we describe the integration of a secure storage on the mobile devices that leverages Trusted Execution Environments in order to safeguard users' credentials on their user device. Next, we provide a general description on how the PABAC infrastructure can be used in application scenarios, namely one for each pilot of the project. The deliverable concludes with the privacy and security considerations that the consortium takes in mind with regard to the developed software and hardware solutions.

## Table of Contents

Executive Summary .....	4
List of Figures .....	7
1 Introduction .....	8
2 P-ABAC Advanced Extensions .....	9
2.1 Attribute Based Encryption .....	9
2.1.1 A multi-authority CP-ABE architecture .....	10
2.1.2 Exploit CP-ABE to prove credential possessions .....	13
3 P-ABAC Module Extensions Implementation .....	15
3.1.1 MA-CP-ABE P-ABAC Scheme .....	16
4 Privacy-preserving Attribute-based Authentication Implementation and Integration .....	17
4.1.1 ABE P-ABAC FIWARE native implementation .....	17
5 Access Control Policies Reasoning Tool .....	19
6 Consent Management .....	27
6.1 Consent Management Reasoning Tool .....	<b>Error! Bookmark not defined.</b>
6.1.1 High Level Operations .....	<b>Error! Bookmark not defined.</b>
6.1.2 Create Policy .....	<b>Error! Bookmark not defined.</b>
6.1.3 View Policy .....	<b>Error! Bookmark not defined.</b>
6.1.4 Delete Policy .....	<b>Error! Bookmark not defined.</b>
6.1.5 Evaluation Requests .....	<b>Error! Bookmark not defined.</b>
6.1.6 Policy Recommendation .....	<b>Error! Bookmark not defined.</b>
6.2 Consent Management Web Front-End .....	<b>Error! Bookmark not defined.</b>
6.2.1 Create Consent Policies .....	<b>Error! Bookmark not defined.</b>
6.2.2 View and Delete Consent Policies .....	<b>Error! Bookmark not defined.</b>
6.2.3 Consent Policies Recommendations .....	<b>Error! Bookmark not defined.</b>
6.3 Consent Management Mobile Front-End .....	<b>Error! Bookmark not defined.</b>
6.3.1 Create new Consent Policy .....	<b>Error! Bookmark not defined.</b>
6.3.2 View and Manage Consent Policies .....	<b>Error! Bookmark not defined.</b>
7 Privacy Awareness .....	48
7.1 De-anonymization Risk Assessment .....	48
7.1.1 De-anonymization Risk Assessment categories .....	49
8 TEE-based Secure Storage .....	51
8.1 Cryptographic Credentials Storage .....	51
8.2 Secure Key Storage .....	52
9 Application Scenarios .....	54

9.1	Support to Financial Services .....	54
9.1.1	Before ABAC .....	55
9.1.2	After ABAC .....	55
9.2	Campus Wi-Fi and Campus-Restricted Web Services .....	56
9.2.1	Before P-ABAC .....	57
9.2.2	After P-ABAC .....	57
9.3	Age Verification .....	58
9.3.1	Before ABAC .....	59
9.3.2	After ABAC .....	60
9.4	ISIC Student Discounts .....	60
9.4.1	Before and After ABAC .....	61
9.5	Federated P-ABAC WiFi (WIFAB) .....	61
9.5.1	Introduction .....	62
9.5.2	Background and Related Work .....	63
9.5.3	Models and Assumptions .....	66
9.5.4	Proposed Approach .....	68
9.5.5	Implementation and Results .....	70
9.5.6	Discussion and Conclusion .....	73
10	Privacy and Security Considerations .....	74
10.1	Federated P-ABAC WiFi .....	74
10.1.1	Passive attacker .....	74
10.1.2	Active attacker .....	75
10.2	Privacy and Security Considerations for the proposed MA-CP-ABE P-ABAC Scheme .....	75
11	Conclusion .....	76
12	References .....	77

## List of Figures

Figure 1 Simple example of issuance .....	13
Figure 2 Protocol description.....	14
Figure 3 CP-ABE protocol description .....	14
Figure 4 P-ABAC components view of the ReCRED architecture.....	15
Figure 5 ReCRED P-ABAC functional architecture with elements involved .....	16
Figure 6 ReCRED P-ABAC software architecture with involved modules .....	16
Figure 7 Create a new Access Control Policy .....	19
Figure 8 Account Based Access Control Policy Creation .....	20
Figure 9 Screen where the user can see and delete the defined ABAC policies .....	21
Figure 10 Screen where the user can see and deleted the defined AccBAC policies.....	21
Figure 11 Screen where the user can specify the criticality of a resource .....	22
Figure 12 Supported operations by the Policy Reasoning tool.....	24
Figure 13 Example of a policy redundancy check .....	<b>Error! Bookmark not defined.</b>
Figure 14 Example of a policy merge check.....	<b>Error! Bookmark not defined.</b>
Figure 15 Example of a partial merge check.....	<b>Error! Bookmark not defined.</b>
Figure 16 Example of the Machine Learning Policy Generator .....	26
Figure 17: Create a policy web interface .....	<b>Error! Bookmark not defined.</b>
Figure 18: View and Delete Consent Policies Interface .....	<b>Error! Bookmark not defined.</b>
Figure 19: Policy Recommendation Web Interface .....	<b>Error! Bookmark not defined.</b>
Figure 20 Mobile app screen for creating new consent policy.....	<b>Error! Bookmark not defined.</b>
Figure 21 List of Consent Policies.....	<b>Error! Bookmark not defined.</b>
Figure 22 View policy details .....	<b>Error! Bookmark not defined.</b>
Figure 23 Delete a consent policy .....	<b>Error! Bookmark not defined.</b>
Figure 24 Cryptographic Credentials Storage on User's Device Protocol Stack .....	51
Figure 25 Cryptographic Credentials Storage overview .....	52
Figure 26 Secure Key Storage overview.....	54
Figure 27 An example of Multi-Authority CP-ABE .....	65
Figure 28 Notation Table .....	67
Figure 29 An example scenario of the proposed approach .....	69
Figure 30 Average times of main cryptographic operations executed on the router (a low-end MIPS device) using the ported Charm framework.....	71
Figure 31 Average key generation and encryption times on the router vs. number of ANDed attributes in policy and ciphertext size vs. number of ANDed attributes in policy .....	72
Figure 32 Required client connection times vs. number of attributes .....	73

## 1 Introduction

The main goal of the ReCRED project is to design and implement an architecture that allows users to simplify identity management by consolidating multiple accounts and by exploiting the concept of *Device Centric Authentication (DCA)*. Another main goal of the project is to go beyond the state-of-the-art of Device Centric Authentication platforms by integrating privacy-aware access control capabilities, which are lacking in actual DCA platforms. Indeed, this is a natural step since user identities are basically formed by user attributes that can be exploited in order to realize an Attribute Based Access Control (ABAC) on resources that the user wants to access. Moreover, since the ReCRED project targets mainly the security of the users' identity and privacy in general, it is needed to add privacy-preserving capabilities to the classical ABAC definition and technologies. This motivates the strong focus of the ReCRED project on the **integration** and **deployment** of state-of-the-art anonymous credentials platforms such as Idemix [33], U-Prove [34], and Attribute Based Encryption (ABE) [21] to realize a Privacy-Preserving Attribute Based Access-Control (P-ABAC) architecture that is able to guarantee the anonymity of the involved users.

The integration of such anonymous credential systems (Idemix and U-Prove), enriched with the state-of-the-art of cryptographic protocols like Attribute Based Encryption (ABE) is realized by means of the FIWARE Privacy Open RESTful API [45]. This enables ReCRED to implement the P-ABAC architecture exposing to the developer a single interface to be exploited for all these protocols. Moreover, our P-ABAC infrastructure is designed to be completely integrated in commercial and widely used authentication systems like FIDO and OpenID connect (OpenAM implementation).

The ABAC architecture will maintain the main device centric approach of the project thanks to the integration of these ABAC systems in the user's device, supported by the deployment and execution inside the trusted execution environment (TEE) on the user-device. To reach such target, the ReCRED project is investigating existing and open source TEE platforms in order to provide a full and secure implementation of ABAC technologies directly on the device.

To complement the aforementioned P-ABAC related functionalities, ReCRED also aims to offer a software suite of tools that will assist end-users in the interactions with the developed P-ABAC systems. Specifically, we offer the following tools:

- **Access Control Policies Reasoning Tool.** It allows Service Provider administrators to easily manage the complex access control policies that are in place. The tool consists of an intuitive user-interface that allows users to create, view, delete, update access control policies. Furthermore, it encapsulates a set of algorithms that aim to provide recommendations to the administrators so that there will be no redundant and obsolete policies. Also, we incorporated a machine learning based recommendation system that aim to recommend new policies. The recommendation engine takes into account a lot of aspects such as the criticality of the resource and the involved user attributes. More details regarding the access control policies reasoning tool can be found in Section 5.
- **Consent Management Tool.** The consent management tool is a very important tool that is made available to various actors of the ecosystem. Specifically, users and identity providers' administrators are able to define policies that define when it is possible to reveal or transfer identity attributes among the entities of the ReCRED ecosystem. For example, a user may not want to reveal a critical identity attribute, such his bank account balance, to entities that



are below a specific level of assurance. The tool also supports a machine learning-based recommendation engine that aims to provide recommendations about new policies to end-users. Specifically, we used recent advancements of deep learning in order to build a recommendation engine that takes into account a wide variety of aspects such as criticality of the identity attributes, information of the entities that are involved in the policies, etc. More details regarding the consent management module can be found in Section 6.

- **Privacy Awareness Tool.** The privacy awareness tool aims to provide insights to end-users regarding the underlying risks that are involved when revealing identity attributes. Specifically, we aim to inform end-users regarding de-anonymization risks when revealing identity attributes. To achieve this, we make use of statistical models that aim to calculate the risk of de-anonymization. For example, we inform the user before revealing identity attributes that based on the already revealed identity attributes a Service Provider may also be able to infer some identity attributes without the explicit reveal of the end-user. This tool is particularly interesting and useful for the user as it enhances the user’s privacy. More details regarding this tool can be found in Section 6.

The rest of this deliverable is organized as follows: In Section 2 we present the details with regard to the developed attribute based encryption solutions and in Section 3 we describe the design and implementation of the developed P-ABAC extensions. Section 4 is dedicated to the privacy preserving attribute based authentication and integration efforts. Section 5 and Section 6 present the access control policies reasoning tool and consent management tool respectively. In Section 7 we describe the privacy awareness tool whereas in Section 8 we discuss the implementation details of the secure storage on users’ device. Section 9 describes the use of the developed infrastructure in the context of the projects’ pilots and Section 10 describes our privacy and security considerations. Finally, the deliverable concludes in Section 11.

## 2 P-ABAC Advanced Extensions

### 2.1 Attribute Based Encryption

We here report an introduction to attribute-based encryption, as already provided in previous WP5 deliverables.

Attribute Based Encryption (ABE) is an emergent new kind of asymmetric cipher. Similar to ordinary public key encryption schemes, a content is encrypted using a public key which does not reveal any information useful to decrypt the data, e.g. the private key. However, unlike ordinary public key schemes, the decryption key is not unique, but multiple users, having different keys, may decrypt the same message. Furthermore, and distinguishing feature of ABE, a user can decrypt a message only if the user is provided with a set of attributes which satisfy a given policy.

We are specifically interested in a variant of ABE called Ciphertext Policy, CP-ABE, where the policy which needs to be satisfied by the user’s attributes is directly integrated in the encrypted data itself, hence it “travels” with the data. Note that the combination of encryption (for confidentiality) and policy (for access control) in CP-ABE appears to be an extremely convenient approach for services that requires to release a specific resource outside of the trusted limits. This is best understood going into an example scenario using CP-ABE as cryptographic technique.

Let us assume that a content provider  $P$  wishes to encrypt a message  $M$  for a given set of users without the need to know a priori the identity of each individual user which shall be able to access the data, but wants to permit access to the data only to users which satisfy a given policy  $\Pi$  expressed in terms of attributes associated to the end users. Such a policy can be any arbitrary combination of “AND” and “OR” conditions, for instance

$$\Pi = (\text{italy:citizen AND job:executive}) \text{ OR } (\text{job:doctor})$$

Notably, at encryption time, the content provider only requires to know:

- the subset of attributes of interest, which are ordinary natural language strings
- the public key of the authority which has issued such attributes (as discussed later, CP-ABE was recently extended to operate with multiple non-coordinating authorities).

Not only CP-ABE does *not* require the content provider to a priori know the set of users which will be able to access the message, but it completely decouples the encryption process from the management of the user attributes. Indeed, suppose that CP-ABE encryption of a message  $M$  using policy  $\Pi$  occurs at a given time, say  $t_1$ . Let  $E[\Pi, M]$  be the resulting ciphertext, where we use a notation which highlights the fact that the policy  $\Pi$  is indeed integrated in the encrypted data itself. Suppose now that, at a subsequent time  $t_2 > t_1$ , a new user, say  $U_x$ , needs to be added to the set of users allowed to access the message. The user just need to retrieve the attributes required to decrypt and, such operation can be performed offline and once-for-all by contacting the related issuing authorities. The data itself does not require modification and continue to travel in the network or on untrusted storage without losing security capabilities.

### 2.1.1 A multi-authority CP-ABE architecture

The concept of CP-ABE has been originally introduced by Bethencourt, Sahai and Waters in 2007 [42]. This first construction however had a significant practical limitation in the fact that attributes were issued by a single, global, authority. In order to overcome such a limitation, the cryptographic community attempted to devise multi-authority CP-ABE schemes, with the first proposal in this field being a paper by Chase [43]. However, this first multi-authority proposal, as well as the subsequent extensions, still required some form of cooperation (at least offline) among the authorities. In the real world, such form of cooperation is deemed to be unviable, as it would force all possible authorities (ranging from banks, governments, visa offices, and even individuals) to interact at least once each other, as well as re-run a cooperation protocol every time a new authority is deployed.

Also, mostly for this reason, CP-ABE did not have any notable practical success outside the restricted community of cryptographic researchers.

In a breakthrough paper, dated 2011 [44], Lewko and Waters proposed the first fully decentralized CP-ABE construction, thus broadly extending CP-ABE’s application range and make it fit the real world needs of large scale networks and deployments. In this context, fully decentralized means that access policies can be specified over an arbitrary set of attributes issued by multiple independent and not cooperating authorities (possibly not even knowing each other’s existence). The far from being trivial technical challenge solved in [44] was the construction of a scheme resistant to collusion among users; in other words, if user  $U_1$  holds attribute  $\text{attr}_1$  issued by an authority  $A_1$  and user  $U_2$  holds attribute  $\text{attr}_2$  issued by a *different* authority  $A_2$  which has never cooperated or exchanged any information with  $A_1$ , and even if the two users collude by exchanging their secrets associated to such attributes, as well as any other possible information locally held by the two users, it should be impossible (computationally hard) for each of these users to decrypt a message encrypted with the policy  $\Pi = (\text{attr}_1 \text{ AND } \text{attr}_2)$ . We refer the reader to the original work [44] for the cryptographic construction details. Despite the original construction, still suffers of some minor technical limitations, we believe that the notion of independent and fully decentralized authority therein exploited very well fits with the real-world needs.

Motivated by the availability of an actual, fully decentralized, multi-authority CP-ABE cryptographic construction, in what follows we preliminary sketch a multi-authority CP-ABE-based security architecture.

**Attribute-issuing authorities.** An authority  $A_i$  is any *arbitrary* entity (hence even including individual users) which autonomously decides to issue attributes. The set-up of an authority is thus an independent decision, and does not require any coordination or interaction with a global authority. The only requirement an authority must adhere is to use a same set of globally-defined and publicly known system parameters (in essence, a small set of standardized parameters, which, to make an illustrative example for the specific CP-ABE setting of [44], appendix D, simply consist in a bilinear group  $\mathbb{G}$  of prime order  $p$ , in a generator  $g$  of the prime order group, and in a hash function  $H$  mapping global identity names into points of the group  $\mathbb{G}$ ). An authority  $x$  will be characterized by a pair of keys: a public key  $A_{x,PK}$ , and an associated secret key  $A_{x,SK}$ . An authority is univocally identified by its public key: since this public key cannot be decided by the authority, but is computed by a cryptographic algorithm, the possibility that two authorities shall have the same PK is negligible. Although not technically necessary, if human readable names shall be used for authorities, an ordinary PKI must be supplementary used to bind the authority’s public key to its real world name, and avoid authority impersonation attacks. More formally, we summarize the setup of an authority with a publicly known algorithm:

$$A_x.AUTHORITY\_SETUP(\text{global parameters}) \rightarrow A_{x,PK}, A_{x,SK}$$

which is independently run by each authority, and which computes the authority’s public and private key pair.

**Attributes.** An attribute is a plain text string defined by, and associated to, an authority. For instance, an attribute can be as general as the string “visa” associated to a country-wide immigration authority and used to grant access permissions to a given country, or as specific as the string “office-mate” issued by an individual. Attributes shall not need to be globally unique (thus simplifying naming issues), but just need to be unique inside a same authority. For example, two countries (say

Russia and Japan) can issue the same attribute string named “visa”, but these two attributes are different as they are issued by different authorities. Whenever ambiguity occurs, we will use the scope symbol “:” to differentiate the two attributes, e.g. Russia:visa versus Japan:visa, but we stress that this is just a notational convenience and not the bit string associated to the actual attribute (which, in both cases, it is simply the string “visa”).

**Attribute-issuing procedure and Global identity names.** In order to get an attribute from an authority, a user must have a global identity name, called UID (user ID), which must be a globally unique bit-string, for instance, an email address. Attributes issued to different identity names (even if belonging to a same human user, e.g. two different email addresses) will not be combined in a same access control policy. For instance, if the same human user holds two identity names, e.g. foo@mail.com holding attribute x and foo@recred.com being issued attribute y, the user will *not* be able to access a data encrypted with CP-ABE using the policy (x AND y). In order to be granted an attribute, a user will offline submit to an authority its global identity name, and if the authority decides to issue the required attribute, the user will receive back a secret key uniquely associated to both the user as well as the attribute. Note that this implies that different users will get different secrets for the same attribute. Formally, we summarize the attribute issuing procedure as an algorithm

$$A_x.\text{ATTRIBUTE\_ISSUING}(\text{UID}, \text{attr\_j}, A_{x,\text{SK}}) \rightarrow K_{\text{UID},\text{attr\_j}}$$

Where UID is the global identity name of the user, attr\_j is the issued attribute name,  $A_{x,\text{SK}}$  is the Authority secret key, and  $K_{\text{UID},\text{attr\_j}}$  is the secret key released to the user for the considered attribute. This algorithm shall be executed by the authority, and the resulting secret key shall be provided to the user via a secure channel.

Note that the compelling aspect of the above sketched architecture resides in the fact that it does not specify any necessary system component (e.g. unlike IPsec, where security associations require to be supported by security association databases and security policy databases). The trust model underlying the access control operation is mandated to individual trust relations (which can eventually, but not necessarily, exploit a certification PKI infrastructure) among entities and attribute-issuing authorities, rather than to a trust infrastructure. This can be very clearly highlighted through the following encryption use-case example. Assume that user  $U_x$  decides to share a message M encrypted with the policy

$$\Pi = (\text{Italy:citizen AND age:greater\_than\_18 AND } U_x:\text{friend}) \text{ OR } (\text{italian\_police:officer})$$

where attributes are written using scope notation (i.e., authority:attribute). In order to encrypt message M, the user needs to decide/have:

- the attribute bit strings, i.e. “citizen”, “greater\_than\_18”, “friend”, and “officer”;
- the access control policy;
- the public keys of the four involved authorities, i.e.
  - a national authority from Italy which releases citizenship permissions;
  - an authority which certifies, by issuing a relevant attribute, that a user has an age greater than 18;
  - a national police authority, and

- the user herself; indeed, since any entity can become authority, the user can as well decide to issue her own attributes, such as the “friend” attribute highlighted in the policy.

Once the message is encrypted, the user knows that the message will be accessed only by other users which have been issued a set of attributes by the specific authorities considered. Indeed, the encryption of a message is performed by ciphering the message using the attribute bit strings as well as the public keys of the relevant authorities which are in charge of issuing the given attributes. Note that this is a significant generalization of the ordinary asymmetric public key encryption, with the notable difference that the public key used during encryption is not anymore, the one of the recipient of the message, but are those of the attribute issuing authorities. In essence, in terms of trust, CP-ABE implies that the user just relies on her individual trust in the *specific* authorities involved, which are identified through their public keys.

Since Attribute Based Encryption schemes realize an implicit access control mechanism on the encrypted data, we believe that the ReCRED P-ABAC architecture could benefit of the usage of such techniques. Indeed, it can be used both to realize an access control on static data distributed in the network (data encryption) both an access control for the user (token encryption).

### 2.1.2 Exploit CP-ABE to prove credential possessions

This section demonstrates the design of a CP-ABE based challenge-response protocol devised to enable users to access resources by proving the possession of secret keys related to a set of attributes. As shown in Figure 1, it is implemented by means of a Challenge-Response message exchange and it is devised to prove the possession of attributes able to satisfy a given boolean policy over such attributes. This simple protocol could be easily implemented and exploited to prove the satisfaction of a policy enforced to dynamic resources by verifying IdPs. This is a complementary approach to anonymous credential systems described in D5.1 and D5.2 that allow to perform the attribute-based access control on resources by means of CP-ABE cryptographic technique described in Section 2.1.1.

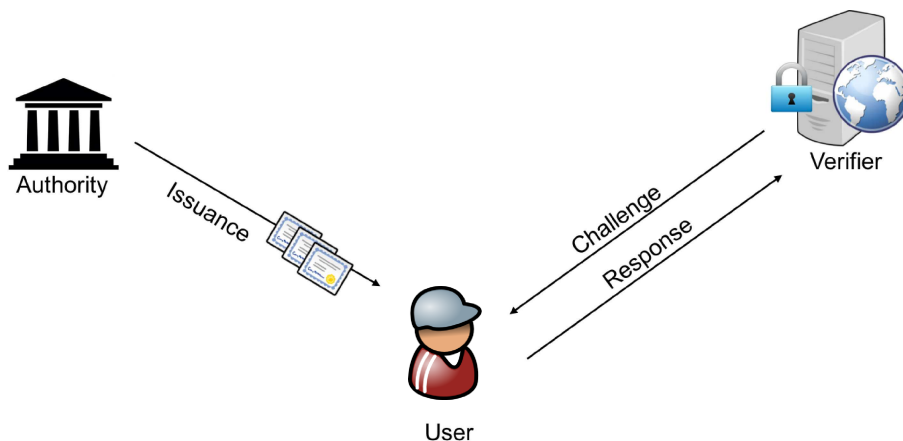


Figure 1 Simple example of issuance

We limit the discussion here to the description of the protocol shown in Figure 2. Indeed, we assume here that the user already has its own set of attributes issued by related authorities, as described in Section 2.1.1. In order to perform a managed access control, the verifying IdP has to specify a policy, over the set of available attributes, to be enforced to the resources it provides and to challenge the user that request the access to the specific resource.

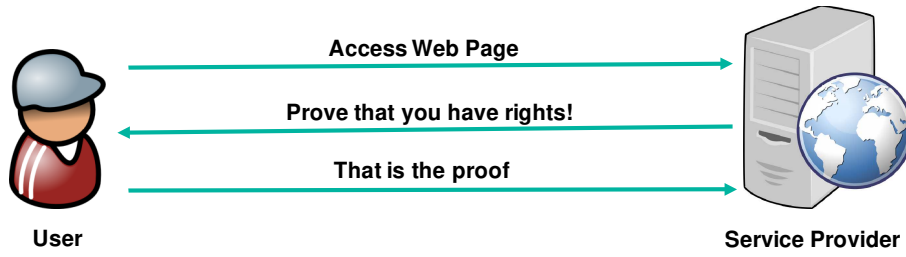


Figure 2 Protocol description

In case of a user requesting access to a resource for which a satisfaction of a policy is required the following protocol (Figure 3) is followed:

- **Verifying IdP:** In order to challenge the user about the possession of attributes required to satisfy the access-control policy it randomly extract a challenge  $C \leftarrow GT$  and encrypt it by means of the CP-ABE encryption algorithm taking as input the plaintext to be encrypted (i.e. the challenge  $C$ ) and the access-control policy  $\Pi$ . The verifying IdP delivers to the user the challenge message containing both the access-control policy  $\Pi$  in cleartext and the CP-ABE encrypted challenge  $\varepsilon(C)$ .
- **User:** After receiving the challenge message, the user should be able to prove to the verifying IdP that she is capable to decrypt the CP-ABE encrypted challenge  $\varepsilon(C)$ . We remind that the user will be able to decrypt the challenge encrypted by means of the policy  $\Pi = \text{boolean\_expression}(attr_0, attr_1, \dots, attr_N)$ , if and only if she owns the private key  $K_{i,GID}$  related to each one of the  $i$  - th attribute satisfying the policy. In the case the user will be able to decrypt the challenge  $C$ , it computes the hash  $C' = H(C)$  of it and sends it to the verifying IdP in order to prove that she was able to access it.

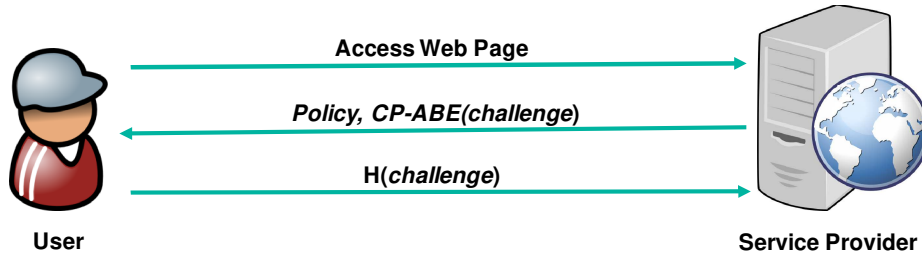


Figure 3 CP-ABE protocol description

To better understand the motivation behind the usage of techniques described in this section, we provide here a simple scenario that demonstrates the convenience in using such approach to perform attribute-based access control on dynamic resources. For ease of presentation we assume that both the CP-ABE infrastructure and the Idemix anonymous credential system are implemented in the scenario. Since we assume that users of the system already own relevant attributes and credentials we are not going to discuss the issuance procedure. Suppose a deployment of a service that provides the access to a database where to read, store and update information needed to correctly make use of the service. It is not possible to use standard CP-ABE to encrypt the database in order to perform the attribute-based access control on it. Indeed, it will be useless to have such service in the environment since it will require each user to download it, perform operations on it a re-upload it on the service. This is clearly not a scalable solution since it allows to perform operations



on it for only one user at each time. On the other side, unlike the plain CP-ABE solution, the proposed challenge-response CP-ABE is suitable for such kind of scenario. By using such technique, the service will be able to enforce the access-control based on attributes of the system and distinguish user accessing the resource (the database) by the owned attributes and the satisfied policies.

### 3 P-ABAC Module Extensions Implementation

We introduce Attribute-Based Encryption (ABE) in ReCRED as an innovative and alternative means to provide Privacy-Preserving Attribute-Based Access Control. To this aim, we provide ABE modules for users, Identity Providers, Service Providers and the Identity Consolidator. The ReCRED architecture allows us to plug-in the ABE stack along with the Idemix and U-Prove stacks, as depicted in Figure 4.

In the user device, the ABE stack is integrated in the Cryptographic Credentials Interface. In the identity providers, the ABE stack is wrapped by the FIWARE Privacy Open RESTful API, as detailed below. In the service providers, we only need minor changes, as the ABE verification engine is located in the Credential Management module of the identity consolidator (which also exposes the FIWARE Privacy Open RESTful API, as described below).

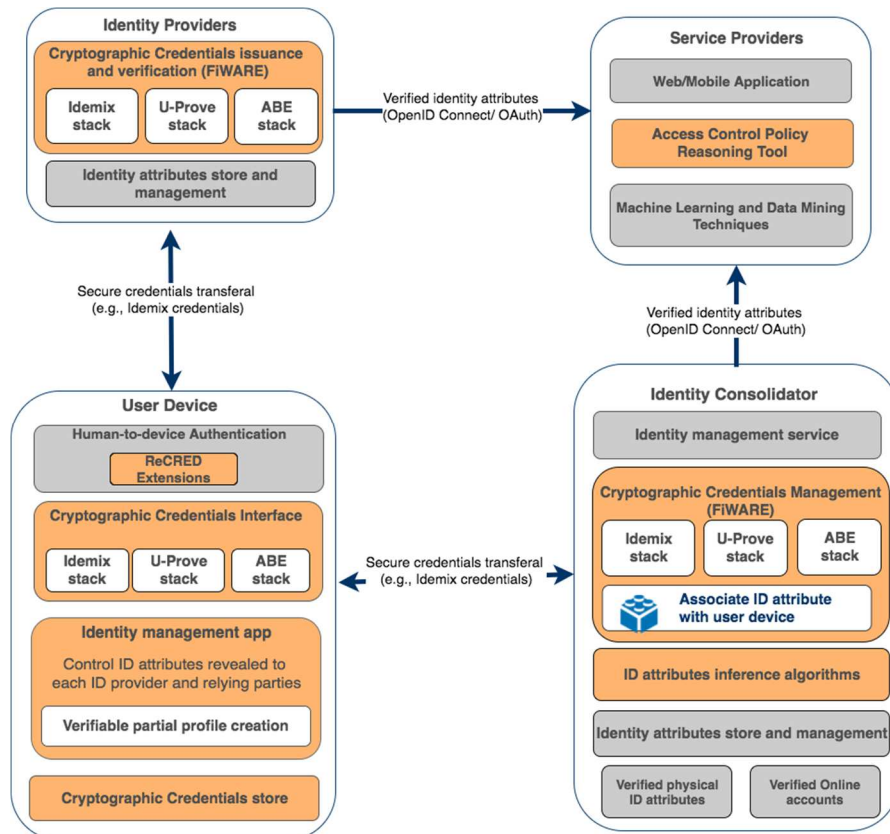


Figure 4 P-ABAC components view of the ReCRED architecture

In particular, we choose to support Multi-Authority Ciphertext-Policy ABE (MA-CP-ABE) [36], as we think it fits naturally into the ReCRED P-ABAC architecture, reported for reference in Figure 4 and Figure 5. Indeed, similar to what happens in the Idemix and U-Prove worlds, MA-CP-ABE allows for

the assignment of attribute-based credentials (MA-CP-ABE keys) from different issuing IdPs (MA-CP-ABE authorities) to users, and on the verifying IdP side, to define access policies based on users’ attributes (MA-CP-ABE encryption). This vision is reflected in the sections below.

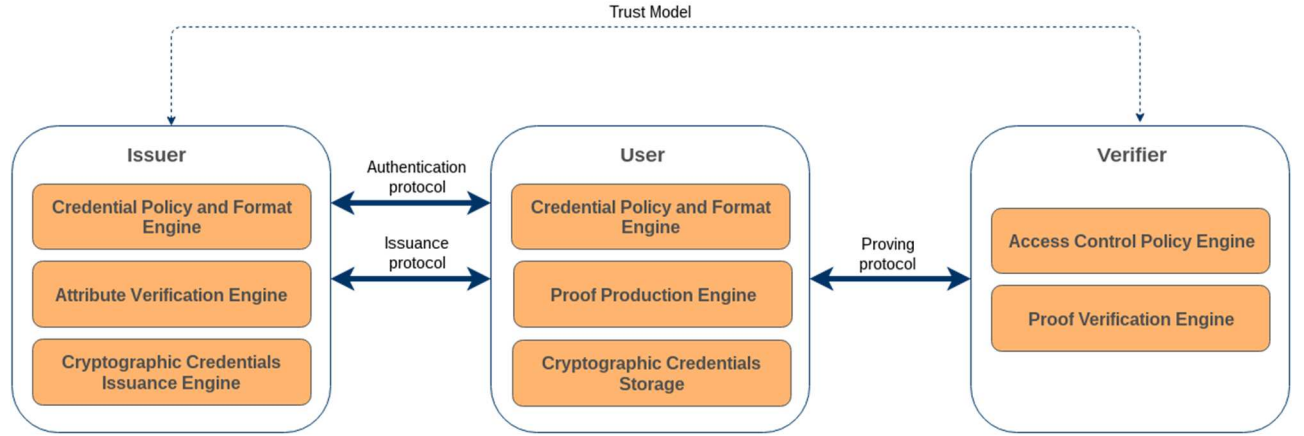


Figure 5 ReCRED P-ABAC functional architecture with elements involved

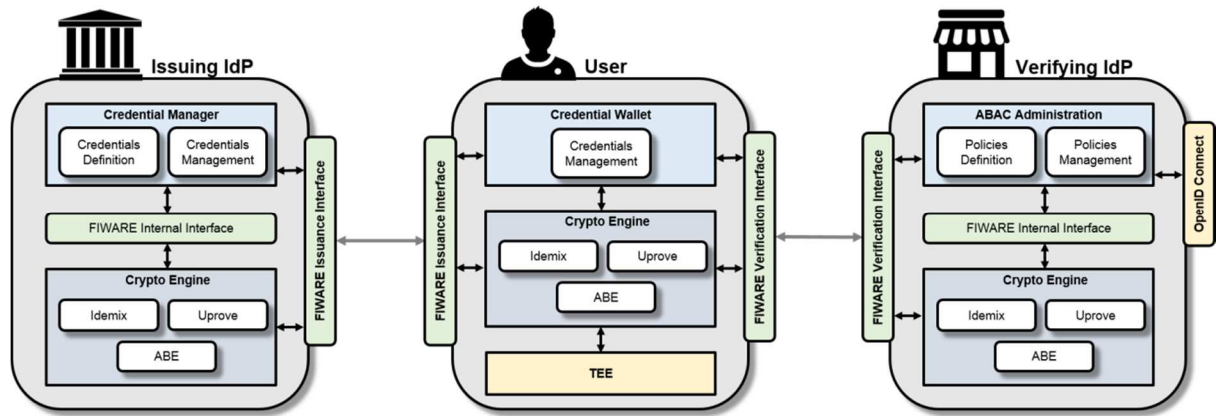


Figure 6 ReCRED P-ABAC software architecture with involved modules

### 3.1.1 MA-CP-ABE P-ABAC Scheme

We here describe a Privacy-Preserving Attribute-Based Access Control (P-ABAC) scheme based on Multi-Authority Ciphertext-Policy Attribute-Based Encryption (MA-CP-ABE) as introduced by [36].

The scheme aims at allowing the definition by service providers of policies for accessing protected resources based on attributes belonging to different domains, while preserving the identity of the users. Compared to Idemix and U-Prove based P-ABAC mechanisms, the proposed scheme allows for a non-interactive, implicit proof of attributes possession through the use of ephemeral secret keys as described below.

#### 3.1.1.1 Setup

In the setup phase, one of the Issuing IdPs generates a set of Global Parameters and publishes them.

All issuing IdPs can then use the Global Parameters to generate a Master Secret Key (MSK), a set of supported attributes and related public keys. Issuing IdPs publish these public keys.



### 3.1.1.2 Issuance

In the issuance phase, the user requests from the Issuing IdP a credential with her attributes. The means of mutual authentication between the user and the issuing IdP and the selection of the attributes to be included in the credential are outside the scope of this scheme.

The Issuing IdP transmits through a secure channel a MA-CP-ABE secret key corresponding to the user attributes.

### 3.1.1.3 Proving

In the Proving/Verification phase, the user requests from the Verifying IdP a resource. The Verifying IdP generates a challenge value to be later used by the User to prove the possession of the attributes. Such a challenge value can be a simple nonce (randomly generated value that SHOULD not be repeated in following authentications) or an ephemeral secret key (a secret key that can be used to access a resource). The challenge is encrypted by the Verifying IdP in accordance to a boolean policy over attributes (that defines the access control rule to the resource). The user that receives the encrypted challenge tries to decrypt it and, in case of success (i.e. she has the required attribute's keys), extracts the value. In case of a nonce value it is required to send it back to the verifying IdP to prove that she was able to access it. In case of an ephemeral secret key (e.g. a WPA/WPA2 PSK) the user will be able to directly use it to access the resource. It will be an implicit proof of possession of the attributes specified in the Boolean policy by the verifying IdP.

## 4 Privacy-preserving Attribute-based Authentication Implementation and Integration

### 4.1.1 ABE P-ABAC FIWARE native implementation

The FIWARE Privacy Open RESTful API [45], described in Section 2.2.2.1 of D5.2, specifies common entities, endpoints and formats for privacy-preserving authentication. APIs of ReCRED modules such as the Identity Consolidator's Credential Manager follow this specification to provide anonymous, yet accountable ABAC, based on the Idemix and U-Prove anonymous credential systems.

As an advanced extension, we provide a **FIWARE-native P-ABAC module based on Multi-Authority Ciphertext Policy Attribute Based Encryption (MA-CP-ABE)** as introduced in [44]. The module allows for the setup of multiple independent ReCRED issuing IdP and allows service providers to define policies based on attributes belonging to different domains.

Note that, given the modularity of P-ABAC architecture defined in D5.1 and finalized in D5.2, it is straightforward to improve the functionalities by pushing additional mechanisms for access control such as MA-CP-ABE. In addition, since the communication protocol follows the FIWARE specification it is just required to adapt the additional messages required by MA-CP-ABE to the right FIWARE format in order to support it.

#### 4.1.1.1 Issuing

The issuance methods defined by the FIWARE Privacy Open RestFUL specification are here specified for MA-CP-ABE.

Method	Description
/issuer/setupSystemParameters/	This method generates the system's Multi-Authority CP-ABE global parameters. The <i>cryptoMechanism</i> parameter should be

	set to the value <i>urn:abc4trust:1.0:algorithm:macpabe</i> .
<b>/issuer/setupIssuerParameters/</b>	This method generates the issuing IdP parameters (based on the MA-CP-ABE global parameters).
<b>/issuer/initIssuanceProtocol/</b>	This method is invoked by the issuing IdP to request the issuance of a MA-CP-ABE private key.
<b>/issuer/issuanceProtocolStep/</b>	At the moment this method is not used but in the future, it could be employed by the user to prove the possession of the issued attributes.

#### 4.1.1.2 Presentation

The presentation (proving) methods defined by the FIWARE Privacy Open RestFUL specification are here specified for MA-CP-ABE.

The steps at the beginning of the presentation phase are here reported<sup>1</sup>:

- the user requests a resource from a service provider;
- the service provider contacts the verifying IdP providing the access policy, and obtains, in return, a session id and a MA-CP-ABE encrypted nonce;
- the service provider sends to the user the encrypted nonce, the session id, a *PresentationPolicyAlternatives* artifact and the URI of the verifying IdP;
- the user decrypts the nonce, to prove the possession of the attributes required by the policy, and generates a *PresentationTokenAlternativesAndPresentationToken* artifact containing the session id and the decrypted nonce.

Then the user, through the supplied URI, invokes the */verification/verifyTokenAgainstPolicy/* method described below. The service provider can then check with the verifying IdP, through the session id, whether the verification has been successful.

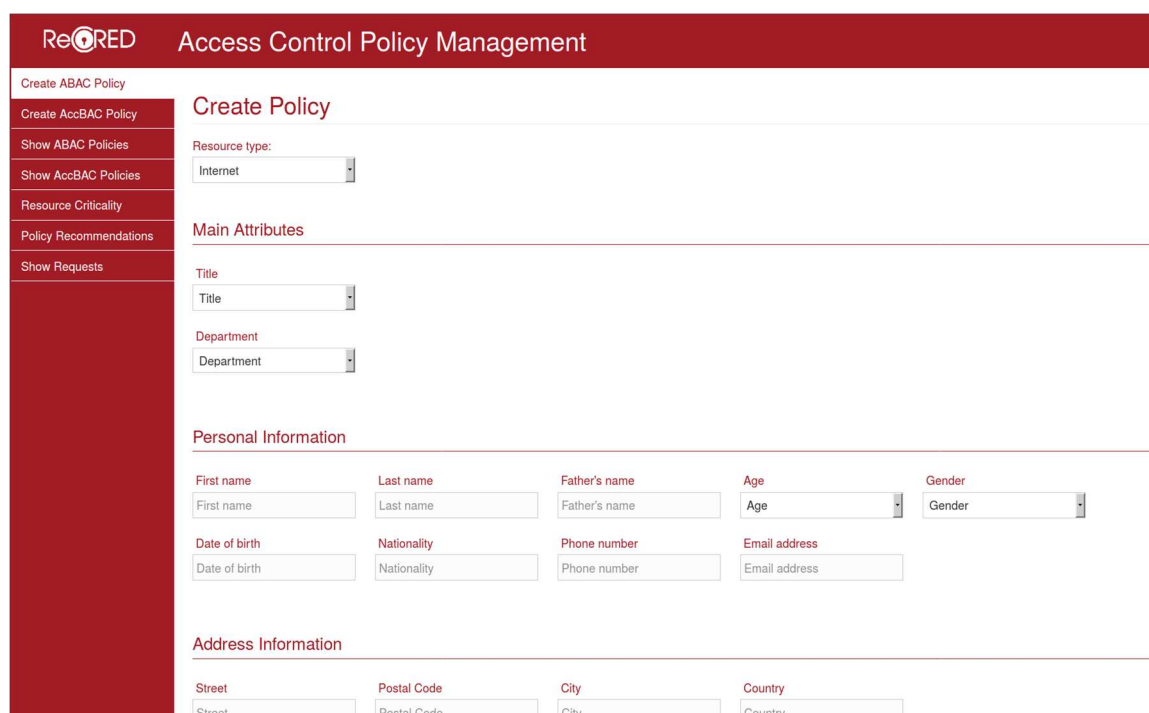
Method	Description
<b>/verification/verifyTokenAgainstPolicy/</b>	This method checks if the decrypted nonce is valid. If it is, it generates, stores and returns a <i>PresentationTokenDescription</i> artifact.
<b>/verification/getToken/</b>	This method looks up a previously verified presentation token. A session id can be supplied.

<sup>1</sup> although these are considered outside the scope of the referenced FIWARE specification

## 5 Access Control Policies Reasoning Tool

The Access Control Policies Reasoning Tool allow Service Provider administrators to setup and manage the access control policies that regulate the access of users to Service Provider specific resources. It consists of a Web interface that allows administrators to easily manage the policies (create, read update and delete operations), a database to store the policies, an XACML-compliant policy decision point and recommendation engine. Below we describe in more detail the components. Note that this Section extends the description of the details that were included in Deliverable 5.2 Section 3.1.10.

**Web Interface for access control policies Management.** The network administrator is presented with a web interface that allow him to create, delete, update and read the existing access control policies. It also enables the user to define the criticality of each resource, which is used by the policy recommendation engine in order to deliver more accurate recommendations based on the criticalities of the resources. Below we provide examples of the interface that demonstrate how a network administrator can perform the aforementioned functionalities.

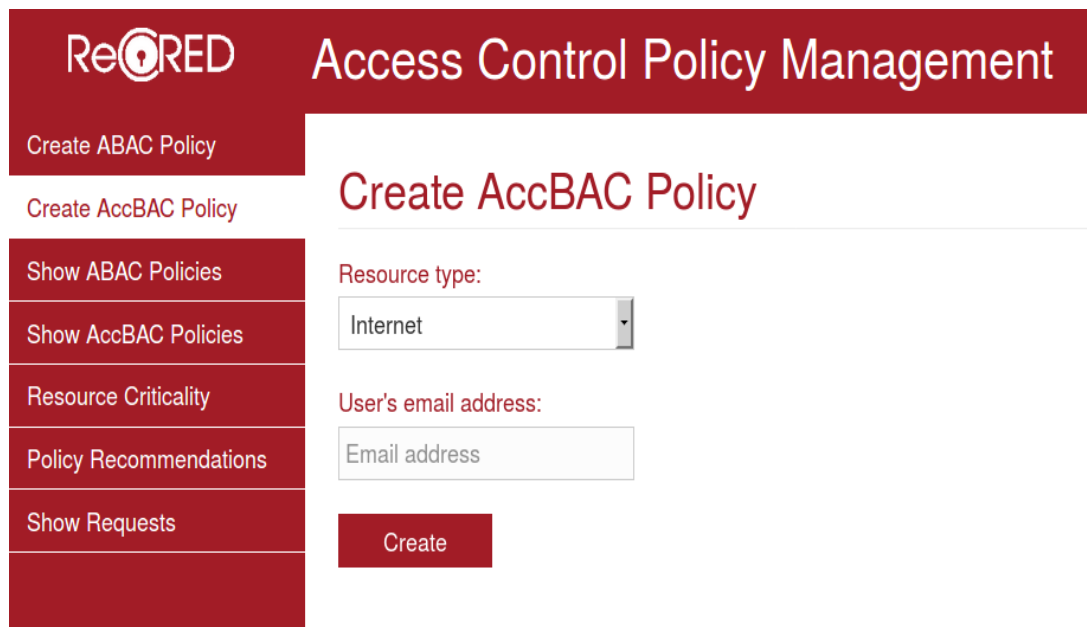


**Figure 7 Create a new Access Control Policy**

Figure 7 demonstrates the access control policy creation screen (attribute-based access control scenario). The user initially selects the type of resource that he wants to create a policy for as well as all the identity attributes that are involved and the desired values that will grant access to the user. When the user submits this form, the system checks if the policy already exists (i.e., a policy with the same identity attributes, values and resource exists) and if not, it stores the access control policy to the database.

In cases where the network administrator wants to grant specific resources to a specific user we also support the creation of control policies that are customized for a specific user (account based access control policy). Figure 8 demonstrates the presented screen when a user wants to create a new account based access control policy. The network administrator only selects the desired resource

and the email of the user that he wants to grant access to. Upon submission of the form, the system checks if the same policy already exists and if not then it stores it to the access control policies reasoning tool database.



The screenshot shows the ReCRED web application interface for 'Access Control Policy Management'. On the left is a dark red sidebar with white text links: 'Create ABAC Policy', 'Create AccBAC Policy', 'Show ABAC Policies', 'Show AccBAC Policies', 'Resource Criticality', 'Policy Recommendations', and 'Show Requests'. The main content area has a dark red header with the ReCRED logo and the title 'Access Control Policy Management'. Below the header, the page title is 'Create AccBAC Policy'. The form contains two input fields: 'Resource type:' with a dropdown menu showing 'Internet', and 'User's email address:' with a text input field containing 'Email address'. A red 'Create' button is at the bottom right of the form.

**Figure 8 Account Based Access Control Policy Creation**

Figure 9 and Figure 10 demonstrate the screens where a user can see and delete the defined access control policies. Specifically, Figure 9 is about the ABAC policies whereas Figure 10 is about the Account based policies. The screens demonstrate the policy number, which is an internal number for housekeeping, the involved resource as well all the identity attribute values that are required. The user is also able to easily delete the policies by clicking the delete button. After user's explicit confirmation the system can delete the policy and also removes it from the database. The same principles apply for the account based access control policies.

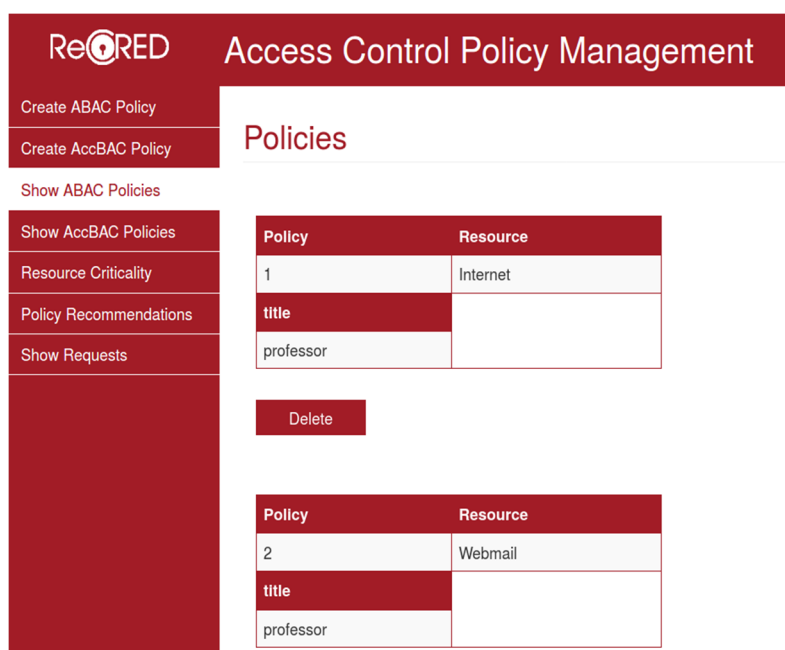


Figure 9 Screen where the user can see and delete the defined ABAC policies

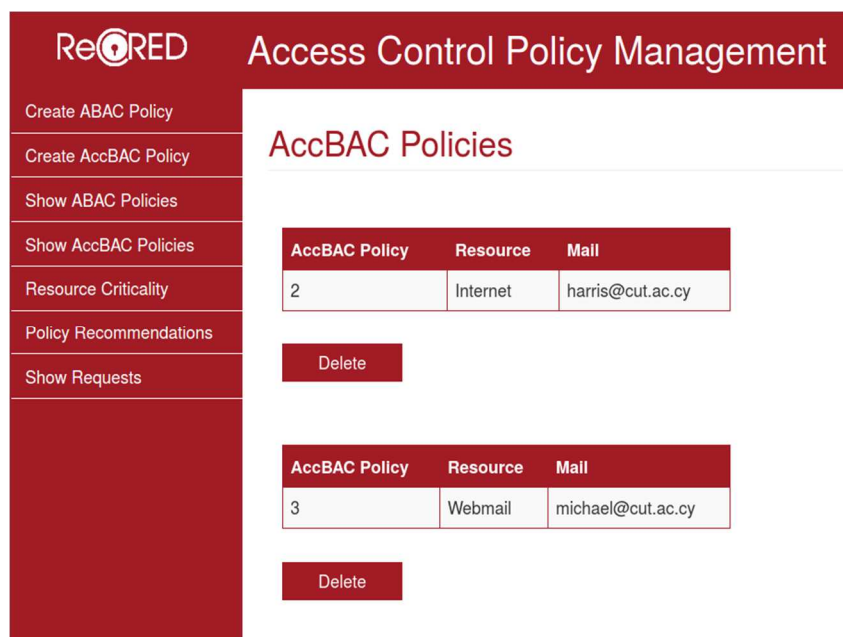
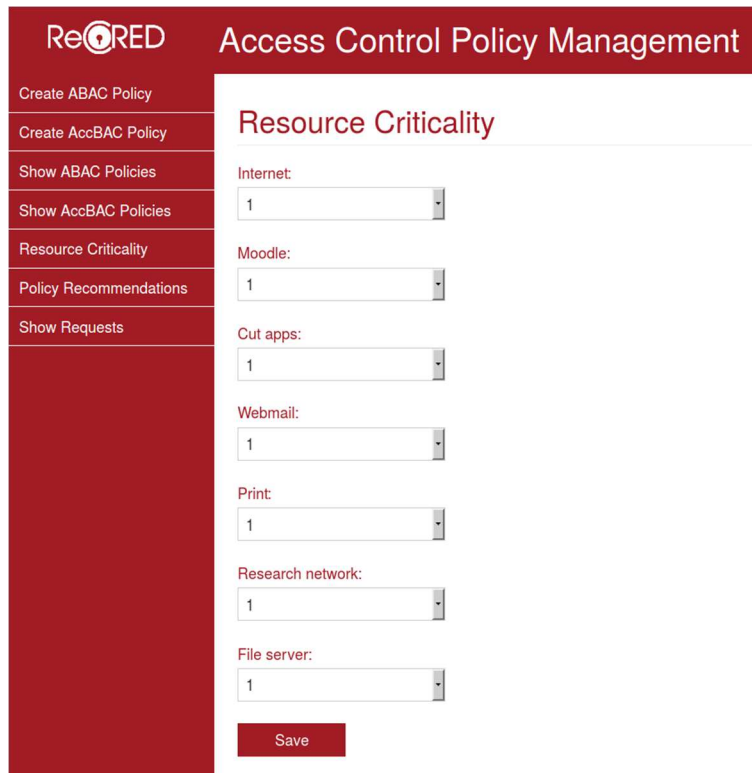


Figure 10 Screen where the user can see and deleted the defined AccBAC policies

The network administrator should also be able to specify the criticality level of a resource. This is particularly important as it allows the machine learning recommendation engine to take this field into account when making recommendations. Figure 11 demonstrates the simple screen where a user can select the desired resource and the criticality level. The criticality levels are between the numbers of 1 and 4, with 1 meaning the highest critical resource whereas 4 means the least critical resource.



The screenshot shows the ReCoRED web interface. On the left is a dark red sidebar with a menu containing: 'Create ABAC Policy', 'Create AccBAC Policy', 'Show ABAC Policies', 'Show AccBAC Policies', 'Resource Criticality' (highlighted), 'Policy Recommendations', and 'Show Requests'. The main content area has a red header with 'ReCoRED' and 'Access Control Policy Management'. Below the header, the title 'Resource Criticality' is displayed. The form contains several dropdown menus, each with the value '1' selected: 'Internet:', 'Moodle:', 'Cut apps:', 'Webmail:', 'Print:', 'Research network:', and 'File server:'. At the bottom of the form is a red 'Save' button.

Figure 11 Screen where the user can specify the criticality of a resource

**Policy Decision Point (PDP).** The PDP is the reasoning tool’s core function which evaluates the incoming requests based on the network administrator’s policies. The whole process is based on the XACML standard, which is an attribute-based access control policy language. Following is a typical example of the PDP evaluation.

First the request is received in a JSON format and is transformed to XACML acceptable JSON format.

An example of the request is the following.

```
{
  "Request": {
    "Action": {
      "Attribute": {
        "AttributeId": "action-id",
        "Value": "access"
      }
    },
    "Resource": {
      "Attribute": {
        "AttributeId": "resource-id",
        "Value": "webmail"
      }
    },
    "AccessSubject": {
      "Attribute": [
        {
          "AttributeId": "department",
          "Value": "electrical engineering"
        },
        {
          "AttributeId": "title",
          "Value": "student"
        }
      ]
    }
  }
}
```

Then the request is evaluated against the policies that are collectively stored in an XACML format and the PDP returns the request decision (Permit/Denied). An example of a Policy is the following.

```
<Policy>
  <Target>
    <AnyOf>
      <AllOf>
        <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">Internet</AttributeValue>
          <AttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id" Category="urn:oasis:names:tc:
        </Match>
      </AllOf>
    </AnyOf>
  </Target>
</Rule>
<Target>
  <AnyOf>
    <AllOf>
      <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">professor</AttributeValue>
        <AttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:accesssubject:title" Category="urn:oasis:names:tc:
      </Match>
    </AllOf>
  </AnyOf>
  <AnyOf>
    <AllOf>
      <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">student</AttributeValue>
        <AttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:accesssubject:title" Category="urn:oasis:names:tc:
      </Match>
    </AllOf>
  </AnyOf>
  <AllOf>
    <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
      <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">Electrical Engineering</AttributeValue>
      <AttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:accesssubject:department" Category="urn:oasis:nam
    </Match>
  </AllOf>
</AnyOf>
</Target>
</Rule>
</Policy>
```


**Database.** The Access Control Policy Reasoning Tool uses a database to store information on policies and requests. The policies are stored in order to properly manage them through the access control management module. And the requests are stored in order to keep a log of the requests but also to be used in the policy recommendation system. Following are examples of the database tables.

policies																		
#	id_abac_policies	resource	firstname	lastname	age	nationality	religion	country	city	phone	email	title	department	yearofstudy	semester	teachingyears	scholarship	
1	7	internet	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	professor	NULL	NULL	NULL	NULL	NULL	NULL
2	8	webmail	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	professor	NULL	NULL	NULL	NULL	NULL	NULL
3	9	moodle	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	professor	NULL	NULL	NULL	NULL	NULL	NULL
4	10	research network	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	professor	NULL	NULL	NULL	NULL	NULL	NULL
5	11	cut apps	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	professor	NULL	NULL	NULL	NULL	NULL	NULL
6	12	file server	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	professor	NULL	NULL	NULL	NULL	NULL	NULL
7	13	print	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	professor	NULL	NULL	NULL	NULL	NULL	NULL
8	14	internet	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	student	Electrical Engineer...	NULL	NULL	NULL	NULL	NULL
9	15	webmail	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	student	Electrical Engineer...	NULL	NULL	NULL	NULL	NULL
10	16	moodle	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	student	Electrical Engineer...	NULL	NULL	NULL	NULL	NULL
11	17	research network	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	student	Electrical Engineer...	NULL	NULL	NULL	NULL	NULL
12	18	cut apps	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	student	Electrical Engineer...	NULL	NULL	NULL	NULL	NULL
13	19	File Server	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	student	Electrical Engineer...	NULL	NULL	NULL	NULL	NULL
14	20	print	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	student	Electrical Engineer...	NULL	NULL	NULL	NULL	NULL



#	log_id	attribute_name	attribute_value
17	6	response_abac	permit
18	6	title	student
19	7	department	electrical engineering
20	7	response_abac	permit
21	7	title	student
22	8	department	electrical engineering
23	8	response	denied
24	8	title	student
25	9	department	electrical engineering
26	9	response_abac	permit
27	9	title	student
28	10	department	electrical engineering
29	10	response	denied
30	10	title	student
31	11	department	electrical engineering

Thus when a request is made, the information is logged. For easier management the network administrator can observe the most recent events at the show requests tab.



## Access Control Policy Management

- Create ABAC Policy
- Create AccBAC Policy
- Show ABAC Policies
- Show AccBAC Policies
- Resource Criticality
- Policy Recommendations
- Show Requests

Response_abac	Permit
Title	Student

Log	Resource
508	Internet
Attribute Name	Attribute Value
Department	Eceei
Response_abac	Permit
Title	Student

Log	Resource
509	Internet
Attribute Name	Attribute Value
Age	23-29
Response_abac	Permit
Scholarship	Yes
Title	Student

Figure 12 Supported operations by the Policy Reasoning tool

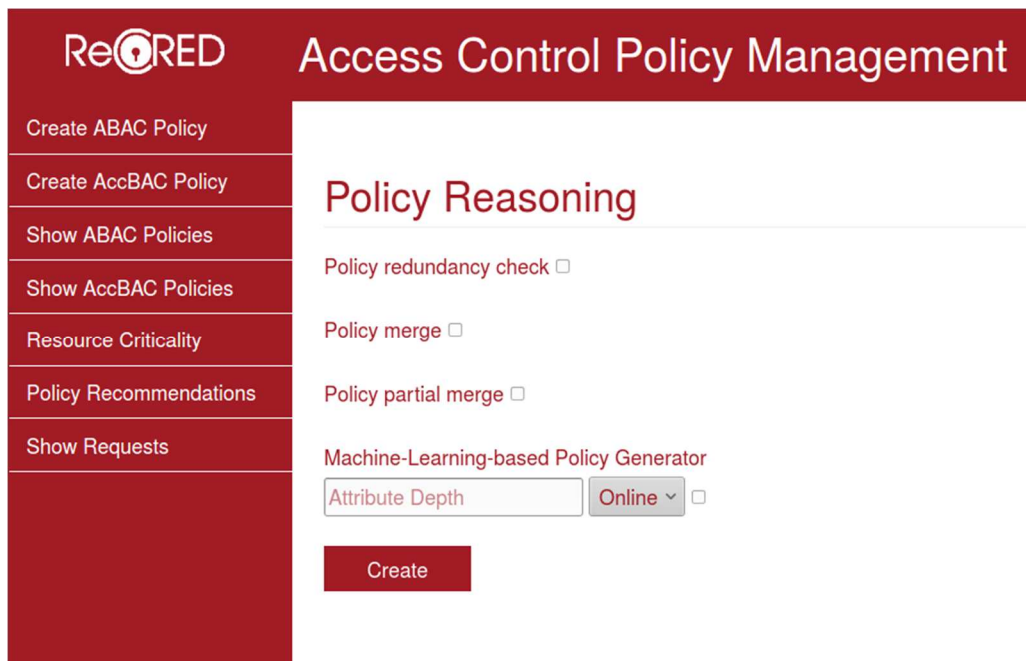
**Recommendation Engine.** In the Policy Recommendations tab there are a few functions to help the user manage the policies. Specifically we have:

- **Policy Redundancy check:** It checks if the service provider administrator's new policies that were deployed make older policies obsolete. This is different from the check when a new policy is created. If such a case exists, the system will redirect the user to a new webpage where the obsolete policies and the policy that makes them obsolete will be presented to the user. Here we require a confirmation from the user to make the appropriate changes,



i.e. delete the obsolete policies. Like in the previous cases, a notification will be prompted to inform the user that the changes have been applied.

- **Policy Merge Check:** The second function checks if any set of ABAC policies can be merged to one. This is the case when we have attributes that have a discrete number of possible values. If such a case exists, the system will redirect the user to a new webpage where the policies to be merged and the new merged policy will be presented to the user. Here we require a confirmation from the user to make the appropriate changes i.e. delete the set of policies and create the new one. Like in the previous cases a notification will be prompted to inform the user that the changes have been applied
- **Policy Partial Merge Check:** The third function is an adaptation of the second one where we don't require all the discrete number of possible values of an attribute to be present i.e. one can be missing. If such a case exists, the system will redirect the user to a new webpage where the policies to be merged and the new merged policy will be presented to the user. Here we require a confirmation from the user to make the appropriate changes i.e. delete the set of policies and create the new one. Like in the previous cases a notification will be prompted to inform the user that the changes have been applied. The final decision is up to the network administrator on whether he will proceed with the recommendation.
- **Machine Learning Based Policy Generator:** Where we use Markov Logic Network to provide policy recommendations. The Attribute Depth parameter defines the number of attributes to be included in the policy.



The screenshot shows the ReCRED Access Control Policy Management interface. On the left is a dark red sidebar with the ReCRED logo and a list of navigation items: Create ABAC Policy, Create AccBAC Policy, Show ABAC Policies, Show AccBAC Policies, Resource Criticality, Policy Recommendations, and Show Requests. The main content area has a dark red header with the title 'Access Control Policy Management'. Below the header, the section 'Policy Reasoning' is displayed. It contains four options, each with a checkbox: 'Policy redundancy check', 'Policy merge', 'Policy partial merge', and 'Machine-Learning-based Policy Generator'. The 'Machine-Learning-based Policy Generator' option is selected. Below this option, there is a text input field labeled 'Attribute Depth' and a dropdown menu labeled 'Online' with a checkbox. At the bottom of the form is a red 'Create' button.

Figure 13. Access Control Policy Management Policy Recommendations

The fourth function uses a Markov Logic Network (MLN) to provide recommendations for new policies. The MLN is a machine learning algorithm that provides weights of importance for the policies. Thus when trained the policies with high values are more probable to be accepted by the administrator as recommendations. The MLN is trained based on the existing policies and the request logs, hence the recommendations will be improving at the WiFi pilot is being used.

ReCRED
Access Control Policy Management

Create ABAC Policy
Create AccBAC Policy
Show ABAC Policies
Show AccBAC Policies
Resource Criticality
Policy Recommendations
Show Requests

Resource			
Research network			
title	gender	age	semester
staff	Male	23-29	Spring

Create

Resource			
Moodle			
title	gender	age	scholarship
staff	Female	23-29	Yes

Create

Figure 14 Example of the Machine Learning Policy Generator

The underlying technical aspects of this implementation are the same for both the Access Control Policies Reasoning tool and the Consent Management tool. Therefore, more details with regard to the machine learning based recommendation engine can be found in the next section. The training and (policy) inference process of the MLN is computationally demanding, thus we resort to an offline process. The policy recommendations and their associate weights are computed in weekly intervals or when enough requests are made to render the re-training essential. In addition, online training is not necessary, since the impact of a single request to the weight is relative to the total requests, thus retraining constantly is waste of computational resources.

Furthermore, before training the model the request logs must be properly prepared. To achieve this we first omit the denied requests and then we transform the data with binary one-hot encoding. This encoding is reversed and used at the output of the model to transform the values to the attributes that consists the recommended policy. To keep the recommendations relevant before we transform the data we omit or adapt attributes that are not useful to recommend policies with. For example we omit attributes like “address” and adapt attributes like age from 1-100 values to N age ranges (e.g. <18, 18-25).

## 6 Consent Management

In this section we describe in more detail the consent management module. This section is based on the description that exists in Section 4 of deliverable 4.3. We elected to also include this description on this deliverable too, as the basic primitives of the consent management tool overlap with the primitives of WP5.

The Consent Management Module is a tool to properly define and utilize consent policies on user attributes. These are needed to be evaluated when request for transfer is made. In order to achieve that, the tool is divided into three components. The Back-end that manages policies and requests, a mobile front-end and a web front-end for the user to easily define his/her consent policies of its attributes.

The policies are stored in xml format under the XACML protocol and also at the database for easier management. When a request is made to transfer an attribute from on idp (source) to another (destination), a check is performed in the back-end to evaluate the request. The evaluation checks, a) if the idp (source) gives its consent to transfer the attribute, and b) if the user allows the transfer also. Only, if it passes both the checks the transfer is valid for execution.

The module also provides policy recommendations to the user. The functionality makes recommendations based on the collective amount of active policies of all users and the request logs for transfer requests. These are utilized to train a Markov Logic Network (MLN). The MLN is a machine learning model that infers the weights of importance of policies. Thus when trained the recommended policies (the ones with high valued weights) are essentially the most probable ones. Furthermore, the recommendations are improved as more users are on the platform by retraining the network when enough data is present.

### 6.1 Consent Management Back-End

The Back-End includes a collection of REST API functions that can be grouped into to three categories.

- **User Policy REST operations:** It allows the users to manage policies that define: (i) which identity attributes can be revealed to specific Service Providers and (ii) which identity attributes can be transferred between specific identity providers. For example, the user may define that it does not wish to reveal his/her address to Identity Providers with an Identity Assurance Level (IAL) below a given threshold.
- **Identity Providers REST operations:** It allows Identity Providers to manage policies that define whether specific attributes can be revealed to Relying Parties or whether identity attributes can be transferred among specific Identity Providers or whether the Identity Provider can issue cryptographic credentials that involve specific identity attributes. For example, policies can be in the following form: (i) be able to transfer attribute A to IDP or SP or IDC and (ii) be able to Issue credentials for attribute A with protocol Z. The former is used from IDPs who do not wish certain attributes, or attributes with specific IAL or Authenticator Assurance Level (AAL), to be revealed to certain unauthorized parties or other entities that allow authentication below a certain IAL. For example, the Social Security Administration (ID provider) provides the social security number that should be revealed only to SPs that have high authentication assurances, such as banks. The latter form is used, when an IDP decides that it does not want the attributes of its users to be proven using Idemix/U-Prove and that

they should be proven through the IDP via OAuth instead (so that the IDP always knows where these credentials have been shown). It also allows the IDC to manage consent policies similarly to the Identity Providers.

- **Evaluation REST operations:** Based on the consent policies defined by the users, IDPs and IDC we evaluate requests to transfer attributes or issue credentials. When a request is made we draw the relevant consent policies from the database and create the up to date XACML files in order to evaluate the request. This is the Policy Decision Point under the XACML terminology.

Below we describe in more details the supported REST operations of the Consent Management Module.

### 6.1.1 High Level Operations

The high level operations that the current version of the Access Control Reasoning Tool for Consent Management supports can be divided to the following categories:

1. User Policy REST operations
  - a. Create a Blacklist user policy for an identity provider
  - b. Create a Blacklist user policy for a service provider
  - c. Create a Whitelist user policy for an identity provider
  - d. Create a Whitelist user policy for a service provider
  - e. View a Blacklist user policy
  - f. View a Whitelist user policy
  - g. Delete a Blacklist user policy
  - h. Delete a Whitelist user policy
  - i. View all users Blacklist policies
  - j. View all users Whitelist policies
2. Identity Providers REST operations
  - a. Create a Blacklist identity provider policy for another identity provider
  - b. Create a Blacklist identity provider policy for a service provider
  - c. Create a Blacklist identity provider policy for issuing credentials
  - d. Create a Whitelist identity provider policy for another identity provider
  - e. Create a Whitelist identity provider policy for a service provider
  - f. Create a Whitelist identity provider policy for issuing credentials
  - g. View a Blacklist identity provider policy
  - h. View a Whitelist identity provider policy
  - i. Delete a Blacklist identity provider policy
  - j. Delete a Whitelist identity provider policy
  - k. View all identity providers Blacklist policies
  - l. View all identity providers Whitelist policies
3. Evaluation REST operations
  - a. Evaluate request to transfer attributes
  - b. Evaluate request to issue credentials

### 6.1.2 Create Policy

**Description:** This REST operation is used to create a Policy. For example, user with id 3 wants to deny transfer of attribute surname to identity provider facebook.

**Operation:** POST

[http://consolidator.recred.eu/idc\\_consent\\_management/create\\_policies/<creator\\_type>/<list\\_type>](http://consolidator.recred.eu/idc_consent_management/create_policies/<creator_type>/<list_type>)

### Request:

```
POST
http://consolidator.recred.eu/idc_consent_management/create_policies/<creator_type>/<list_type>

Accept: application/json

Authorization: Bearer <access_token>
```

### Description of Elements in Request URI:

Element	Description	Valid Value
creator_type	The creator of the policy	[user, idp]
list_type	The type of the policy to be created	[blacklist, whitelist]

Following are 2 examples including all creator type options available on the request url

### CREATE A USER POLICY

### Request:

```
POST http://consolidator.recred.eu/idc_consent_management/create_policies/user/<list_type>

Accept: application/json

Authorization: Bearer <access_token>
```

### Description of Elements in Request URI:

Element	Description	Valid Value
list_type	The type of the policy to be created	[blacklist, whitelist]

### Request Body:

```
{
  "user_id":3,
  "idp_a":"twitter",
  "attr_name":"surname",
  "AAL_attr":"2",
  "AAL_attr_func":"less-than-or-equal",
  "IAL_attr":"1",
  "IAL_attr_func":"less-than-or-equal",
  "attr_cs":"1",
  "attr_cs_func":"greater-than-or-equal",
  "idp_b":"facebook",
  "AAL_idp_b":"2",
  "AAL_idp_b_func":"less-than-or-equal",
  "IAL_idp_b":"2",
  "IAL_idp_b_func":"less-than-or-equal",
  "exp_date":"15-2-18" }
```

### Response Body:

```
200
```

```
Content-Type: application/json
{
  "Success": {
    "Created": "Policy",
    "creator_type": "user",
    "list_type": "blacklist"  }}

```

#### Description of Elements in Request Body

Element	Description	Required	Valid Value
user_id	User's unique Identifier	Yes	String
idp_a	The identity provider that is the source of the attribute	Yes	String
attr_name	The name of the attribute	*	String
AAL_attr	The Authenticator Assurance Level of the attribute must have	*	[1,2,3]
IAL_attr	The Identity Assurance Level of the attribute must have	*	[1,2,3]
AAL_attr_func	The function to execute on the level of Authenticator Assurance of the attribute. Requires AAL_attr attribute. If AAL_attr_func not specified default function is equal.	No	[greater-than-or-equal, less-than-or-equal]
IAL_attr_func	The function to execute on the level of Identity Assurance of the attribute. Requires IAL_attr attribute. If IAL_attr_func not specified default function is equal.	No	[greater-than-or-equal, less-than-or-equal]
attr_cs	The confidence score that the attribute must have	*	[0-100]
attr_cs_func	The function to execute on the confidence score of the attribute. Requires attr_cs attribute. If attr_cs_func not specified default function is equal.	No	[greater-than-or-equal, less-than-or-equal]
idp_b	The identity provider that is the destination of the attribute	**	[greater-than-or-equal, less-than-or-equal]
AAL_idp_b	The Authenticator Assurance Level of the identity provider must have	**	[1,2,3]
IAL_idp_b	The Identity Assurance Level of the identity provider must have	**	[1,2,3]
AAL_idp_b_func	The function to execute on the level of Authenticator Assurance of the provider. Requires AAL_idp_b attribute. If AAL_idp_b_func not specified default function is equal.	No	[greater-than-or-equal, less-than-or-equal]
IAL_idp_b_func	The function to execute on the level of Identity Assurance of the provider.	No	[greater-than-or-equal, less-than-or-equal]

	Requires IAL_idp_b attribute. If IAL_idp_b_func not specified default function is equal.		equal]
sp	The service provider that is the destination of the attribute	**	String
exp_date	The day the policy expires	No	Date (YYYY-MM-DD)

\* At least one

\*\* At least one of [idp\_b, AAL\_attr, AAL\_attr] or [sp]

## CREATE AN IDP POLICY

### Request:

```
POST http://consolidator.recred.eu/idc_consent_management/create_policies/idp/<list_type>
Accept: application/json
Authorization: Bearer <access_token>
```

### Description of Elements in Request URI:

Element	Description	Valid Value
list_type	The type of the policy to be created	[blacklist, whitelist]

### Request Body:

```
{
  "idp_a": "twitter",
  "attr_name": "surname",
  "AAL_attr": "2",
  "AAL_attr_func": "less-than-or-equal",
  "IAL_attr": "1",
  "IAL_attr_func": "less-than-or-equal",
  "attr_cs": "1",
  "attr_cs_func": "greater-than-or-equal",
  "idp_b": "facebook",
  "AAL_idp_b": "2",
  "AAL_idp_b_func": "less-than-or-equal",
  "IAL_idp_b": "2",
  "IAL_idp_b_func": "less-than-or-equal",
  "exp_date": "15-2-18" }
```

### Response Body:

```
200
Content-Type: application/json

{
  "Success": {
    "Created": "Policy",
    "creator_type": "idp",
    "list_type": "blacklist"}
}
```

#### Description of Elements in Request Body

Element	Description	Required	Valid Value
idp_a	The identity provider that is the source of the attribute	Yes	String
attr_name	The name of the attribute	*	String
AAL_attr	The Authenticator Assurance Level of the attribute must have	*	[1,2,3]
IAL_attr	The Identity Assurance Level of the attribute must have	*	[1,2,3]
AAL_attr_func	The function to execute on the level of Authenticator Assurance of the attribute. Requires AAL_attr attribute. If AAL_attr_func not specified default function is equal.	No	[greater-than-or-equal, less-than-or-equal]
IAL_attr_func	The function to execute on the level of Identity Assurance of the attribute. Requires IAL_attr attribute. If IAL_attr_func not specified default function is equal.	No	[greater-than-or-equal, less-than-or-equal]
attr_cs	The confidence score that the attribute must have	*	[0-100]
attr_cs_func	The function to execute on the confidence score of the attribute. Requires attr_cs attribute. If attr_cs_func not specified default function is equal.	No	[greater-than-or-equal, less-than-or-equal]
idp_b	The identity provider that is the destination of the attribute	**	[greater-than-or-equal, less-than-or-equal]
AAL_idp_b	The Authenticator Assurance Level of the identity provider must have	**	[1,2,3]
IAL_idp_b	The Identity Assurance Level of the identity provider must have	**	[1,2,3]
AAL_idp_b_func	The function to execute on the level of Authenticator Assurance of the provider. Requires AAL_idp_b attribute. If AAL_idp_b_func not specified default function is equal.	No	[greater-than-or-equal, less-than-or-equal]
IAL_idp_b_func	The function to execute on the level of Identity Assurance of the provider. Requires IAL_idp_b attribute. If IAL_idp_b_func not specified default function is equal.	No	[greater-than-or-equal, less-than-or-equal]
sp	The service provider that is the destination of the attribute	**	String
protocol	The protocol in which to issue credentials	**	
exp_date	The day the policy expires	No	Date (YYYY-MM-DD)



\* At least one

\*\* At least one of [idp\_b, AAL\_attr, IAL\_attr] or [sp] or [protocol]

### 6.1.3 View Policy

**Description:** This REST operation is used to view Policies. For example, I want to view all the blacklist policies of user 5.

**Operation:** GET

`http://consolidator.recred.eu/idc_consent_management/show_policies/<creator_type>/<list_type>/<creator_id>`

**Request:**

```
GET
http://consolidator.recred.eu/idc_consent_management/show_policies/<creator_type>/<list_type>/<creator_id>
Accept: application/json
Authorization: Bearer <access_token>
```

**Description of Elements in Request URI:**

Element	Description	Valid Value
creator_type	The creator of the policy	[user, idp]
list_type	The type of the policy to be created	[blacklist, whitelist]
creator_id	The creator id. If creator_id is not specified it returns the complete list of policies	String

Following are 2 examples including all creator type options available on the request url

## SHOW USER POLICIES

**Request:**

```
GET
http://consolidator.recred.eu/idc_consent_management/show_policies/user/<list_type>/<creator_id>
Accept: application/json
Authorization: Bearer <access_token>
```

**Description of Elements in Request URI:**

Element	Description	Valid Value
list_type	The type of the policy to be created	[blacklist, whitelist]

creator_id	The creator id. If creator_id is not specified it returns the complete list of policies	String
------------	---	--------

#### Response Body:

```

200
Content-Type: application/json
[
  {
    "attr_name": "surname",
    "idp_a": "twitter",
    "idp_b": "facebook",
    "user_id": "3",
    "users_blacklist_id": 10  },
  {
    "attr_cs": "1",
    "attr_cs_func": "greater-than-or-equal",
    "AAL_attr": "2",
    "AAL_attr_func": "less-than-or-equal",
    "IAL_attr": "1",
    "IAL_attr_func": "less-than-or-equal",
    "attr_name": "surname",
    "exp_date": "15-2-2018",
    "idp_a": "twitter",
    "idp_b": "facebook",
    "AAL_idp_b": "2",
    "AAL_idp_b_func": "less-than-or-equal",
    "IAL_idp_b": "2",
    "IAL_idp_b_func": "less-than-or-equal",
    "user_id": "3",
    "users_blacklist_id": 11  }]

```

#### Description of Elements in Response Body

Element	Description	Valid Value
users_blacklist_id	The blacklist unique identifier for the users	Integer
users_whitelist_id	The whitelist unique identifier for the users	Integer
user_id	User's unique Identifier	String
idp_a	The identity provider that is the source of the attribute	String
attr_name	The name of the attribute	String
AAL_attr	The Authenticator Assurance Level of the attribute must have	[1, 2, 3]
IAL_attr	The Identity Assurance Level of the attribute must have	[1, 2, 3]
AAL_attr_func	The function to execute on the level of Authenticator Assurance of the attribute. Requires AAL_attr attribute. If AAL_attr_func not specified default function is equal.	[greater-than-or-equal, less-than-or-equal]
IAL_attr_func	The function to execute on the level of Identity Assurance of the attribute. Requires	[greater-than-or-equal, less-than-or-equal]

	IAL_attr attribute. If IAL_attr_func not specified default function is equal.	
attr_cs	The confidence score that the attribute must have	[0-100]
attr_cs_func	The function to execute on the confidence score of the attribute. Requires attr_cs attribute. If attr_cs_func not specified default function is equal.	[greater-than-or-equal, less-than-or-equal]
idp_b	The identity provider that is the destination of the attribute	String
AAL_idp_b	The Authenticator Assurance Level of the identity provider must have	[1, 2, 3]
IAL_idp_b	The Identity Assurance Level of the identity provider must have	[1, 2, 3]
AAL_idp_b_func	The function to execute on the level of Authenticator Assurance of the provider. Requires AAL_idp_b attribute. If AAL_idp_b_func not specified default function is equal.	[greater-than-or-equal, less-than-or-equal]
IAL_idp_b_func	The function to execute on the level of Identity Assurance of the provider. Requires IAL_idp_b attribute. If IAL_idp_b_func not specified default function is equal.	[greater-than-or-equal, less-than-or-equal]
sp	The service provider that is the destination of the attribute	String
exp_date	The day the policy expires	Date (YYYY-MM-DD)

### SHOW IDP POLICIES

#### Request:

```
GET
http://consolidator.recred.eu/idc_consent_management/show_policies/idp/<list_type>/<creator_id>
Accept: application/json
Authorization: Bearer <access_token>
```

#### Description of Elements in Request URI:

Element	Description	Valid Value
list_type	The type of the policy to be created	[blacklist, whitelist]
creator_id	The creator id. If creator_id is not specified it returns the complete list of policies	String

#### Response Body:

```
200
Content-Type: application/json

[
```

```
{
  "attr_name": "surname",
  "idp_a": "twitter",
  "idp_b": "facebook",
  "idps_blacklist_id": 10 },
{
  "attr_cs": "1",
  "attr_cs_func": "greater-than-or-equal",
  "AAL_attr": "2",
  "AAL_attr_func": "less-than-or-equal",
  "IAL_attr": "1",
  "IAL_attr_func": "less-than-or-equal",
  "attr_name": "surname",
  "exp_date": "15-2-2018",
  "idp_a": "twitter",
  "idp_b": "facebook",
  "AAL_idp_b": "2",
  "AAL_idp_b_func": "less-than-or-equal",
  "IAL_idp_b": "2",
  "IAL_idp_b_func": "less-than-or-equal",
  "idps_blacklist_id": 11 }]
```

#### Description of Elements in Response Body

Element	Description	Valid Value
idps_blacklist_id	The blacklist unique identifier for the identity providers	Integer
idps_whitelist_id	The whitelist unique identifier for the identity providers	Integer
idp_a	The identity provider that is the source of the attribute	String
attr_name	The name of the attribute	String
AAL_attr	The Authenticator Assurance Level of the attribute must have	[1, 2, 3]
IAL_attr	The Identity Assurance Level of the attribute must have	[1, 2, 3]
AAL_attr_func	The function to execute on the level of Authenticator Assurance of the attribute. Requires AAL_attr attribute. If AAL_attr_func not specified default function is equal.	[greater-than-or-equal, less-than-or-equal]
IAL_attr_func	The function to execute on the level of Identity Assurance of the attribute. Requires IAL_attr attribute. If IAL_attr_func not specified default function is equal.	[greater-than-or-equal, less-than-or-equal]
attr_cs	The confidence score that the attribute must have	[0-100]
attr_cs_func	The function to execute on the confidence score of the attribute. Requires attr_cs attribute. If attr_cs_func not specified default function is equal.	[greater-than-or-equal, less-than-or-equal]
idp_b	The identity provider that is the destination of the attribute	String

AAL_idp_b	The Authenticator Assurance Level of the identity provider must have	[1, 2, 3]
IAL_idp_b	The Identity Assurance Level of the identity provider must have	[1, 2, 3]
AAL_idp_b_func	The function to execute on the level of Authenticator Assurance of the provider. Requires AAL_idp_b attribute. If AAL_idp_b_func not specified default function is equal.	[greater-than-or-equal, less-than-or-equal]
IAL_idp_b_func	The function to execute on the level of Identity Assurance of the provider. Requires IAL_idp_b attribute. If IAL_idp_b_func not specified default function is equal.	[greater-than-or-equal, less-than-or-equal]
sp	The service provider that is the destination of the attribute	String
protocol	The protocol in which to issue credentials	String
exp_date	The day the policy expires	Date (YYYY-MM-DD)

#### 6.1.4 Delete Policy

**Description:** This REST operation is used to delete Policies. For example, I want to delete the blacklist policy with id 14.

**Operation:** DELETE

[http://consolidator.recred.eu/idc\\_consent\\_management/delete\\_policies/<creator\\_type>/<list\\_type>/<delete\\_id>](http://consolidator.recred.eu/idc_consent_management/delete_policies/<creator_type>/<list_type>/<delete_id>)

**Request:**

```
DELETE
http://consolidator.recred.eu/idc_consent_management/delete_policies/<creator_type>/<list_type>/<delete_id>
Accept: application/json
Authorization: Bearer <access_token>
```

**Response Body:**

```
200
Content-Type: application/json

{
  "Success": {
    "Deleted": "11",
    "creator_type": "user",
    "list_type": "blacklist"  }}
```

**Description of Elements in Request URI:**

Element	Description	Valid Value
creator_type	The creator of the policy	[user, idp]
list_type	The type of the policy to be created	[blacklist, whitelist]

delete_id	The unique id of the policy. It can be the value of [users_blacklist_id, users_whitelist_id, idps_blacklist_id, idps_whitelist_id]	Integer
-----------	--	---------

### 6.1.5 Evaluation Requests

**Description:** This REST operation is used to evaluate

- 1) The request to transfer attributes based on user’s and identity provider’s Policies
- 2) The request to issue credentials based on identity provider’s policies

**Operation:** POST [http://consolidator.recred.eu/idc\\_consent\\_management/requests/<action\\_type>](http://consolidator.recred.eu/idc_consent_management/requests/<action_type>)

**Request:**

```
POST http://consolidator.recred.eu/idc_consent_management/requests/<action_type>
Accept: application/json
Authorization: Bearer <access_token>
```

**Description of Elements in Request URI:**

Element	Description	Valid Value
action_type	The action to be evaluated	[transfer, issue]

Following are 2 examples including all action type options available on the request url

### TRANSFER REQUEST

**Request:**

```
POST http://consolidator.recred.eu/idc_consent_management/requests/transfer
Accept: application/json
Authorization: Bearer <access_token>
```

**Request Body:**

```
{
  "user_id":3,
  "idp_a":"twitter",
  "attr_name":"surname",
  "AAL_attr":"2",
  "IAL_attr":"1",
  "attr_name_cs":"80.4",
  "idp_b":"facebook",
  "AAL_idp_b":"2",
  "IAL_idp_b":"2" }
```

**Response Body:**

```
200
Content-Type: application/json

{
```

```
"idps_blacklist": "NOT_APPLICABLE",
"idps_whitelist": "APPLICABLE",
"users_blacklist": "NOT_APPLICABLE",
"users_whitelist": "APPLICABLE" }
```

#### Description of Elements in Request Body

Element	Description	Required	Valid Value
user_id	User's unique Identifier	Yes	String
idp_a	The identity provider that is the source of the attribute	Yes	String
attr_name	The name of the attribute	Yes	String
AAL_attr	The level of Authenticator Assurance that the attribute (attr_name) of the user (user_id) has. If this attribute is not given it takes the lowest value (1)	No	[1, 2, 3]
IAL_attr	The level of Identity Assurance that the attribute (attr_name) of the user (user_id) has. If this attribute is not given it takes the lowest value (1)	No	[1, 2, 3]
attr_name_cs	The confidence score that the attribute (attr_name) of the user (user_id) has. If this attribute is not given it takes the lowest value (0)	No	[0-100]
idp_b	The identity provider that is the destination of the attribute	*	String
AAL_idp_b	The level of Authenticator Assurance that the identity provider (idp_b) has. If this attribute is not given it takes the lowest value (1)	No	[1, 2, 3]
IAL_idp_b	The level of Identity Assurance that the identity provider (idp_b) has. If this attribute is not given it takes the lowest value (1)	No	[1, 2, 3]
sp	The service provider that is the destination of the attribute	*	String

\* At least one [idp\_b, sp]

#### Description of Elements in Response Body

Element	Description	Valid Value
users_blacklist	Returns the evaluation of the user's blacklist policies. Returns APPLICABLE if the user denies the request	[APPLICABLE, NOT-APPLICABLE]
users_whitelist	Returns the evaluation of the user's whitelist policies. Returns APPLICABLE if the user permits the request	[APPLICABLE, NOT-APPLICABLE]
idps_blacklist	Returns the evaluation of the identity	[APPLICABLE, NOT-

	provider’s blacklist policies. Returns APPLICABLE if the identity provider denies the request	APPLICABLE]
idps_whitelist	Returns the evaluation of the identity provider’s whitelist policies. Returns APPLICABLE if the identity provider permit the request	[APPLICABLE, NOT-APPLICABLE]

## CREDENTIAL REQUEST

### Request:

```
POST http://consolidator.recred.eu/idc_consent_management/requests/issue
Accept: application/json
Authorization: Bearer <access_token>
```

### Request Body:

```
{
  "idp_a": "twitter",
  "attr_name": "surname",
  "AAL_attr": "2",
  "IAL_attr": "1",
  "attr_name_cs": "80.4",
  "protocol": "uprove" }
```

### Response Body:

```
200
Content-Type: application/json

{
  "idps_blacklist": "NOT_APPLICABLE",
  "idps_whitelist": "APPLICABLE" }
```

### Description of Elements in Request Body

Element	Description	Required	Valid Value
idp_a	The identity provider that is the source of the attribute	Yes	String
attr_name	The name of the attribute	Yes	String
AAL_attr	The level of Authenticator Assurance that the attribute (attr_name) of the user (user_id) has. If this attribute is not given it takes the lowest value (1)	No	[1, 2, 3]
IAL_attr	The level of Identity Assurance that the attribute (attr_name) of the user (user_id) has. If this attribute is not given it takes the lowest value (1)	No	[1, 2, 3]
attr_name_cs	The confidence score that the attribute (attr_name) of the user (user_id) has. If this attribute is not given it takes the lowest value (0)	No	[0-100]



protocol	The protocol in which to issue credentials	Yes	String
----------	--	-----	--------

#### Description of Elements in Response Body

Element	Description	Valid Value
idps_blacklist	Returns the evaluation of the identity provider’s blacklist policies. Returns APPLICABLE if the identity provider denies the request	[APPLICABLE, NOT-APPLICABLE]
idps_whitelist	Returns the evaluation of the identity provider’s whitelist policies. Returns APPLICABLE if the identity provider permit the request	[APPLICABLE, NOT-APPLICABLE]

## 6.2 Consent Management mobile application

The Consent Management mobile front-end is an Android mobile app that allows the users to define their consent for their various identity attributes, by defining policies regarding the Identity Providers and Service Providers to which their attributes should be revealed.

The mobile app communicates with the Consent Management back-end, allowing the end-users to access the following functionality:

- Create new consent policies, which whitelist or blacklist specific identity attributes (or groups of attributes) from Identity Providers and/or Service Providers
- View the consent policies that he has created
- Modify or delete consent policies that he has already created

### 6.2.1 Create new Consent Policy

The user can create new consent policies by defining the following details (Figure 40Error! Reference source not found.):

- **Source Identity Providers:** This is the Identity Provider that maintains the attribute(s) for which the new consent policy will be created.
- **Type of Policy:** The user can select between the following options.
  - Blacklisting Policy, meaning that the selected attributes will always be blocked from the selected IdPs / SPs
  - Whitelisting Policy, meaning that the selected attributes will always be revealed to the selected IdPs / SPs
- **Selected Attribute(s):** The user can select the attribute(s) for which the new consent policies will be created. There are three alternative options.
  - Select a specific attribute,
  - Select many attributes, according to their LoA,
  - Select many attributes, according to their Confidence Score
- **Identity / Service Provider(s):** The user can select the target IdP(s) or SP(s) of the new consent policies. There are four alternative options.
  - Select a specific Identity Provider,
  - Select many Identity Providers, according to their LoA,

- Select a specific Service Provider,
- Select many Service Providers, according to their LoA,

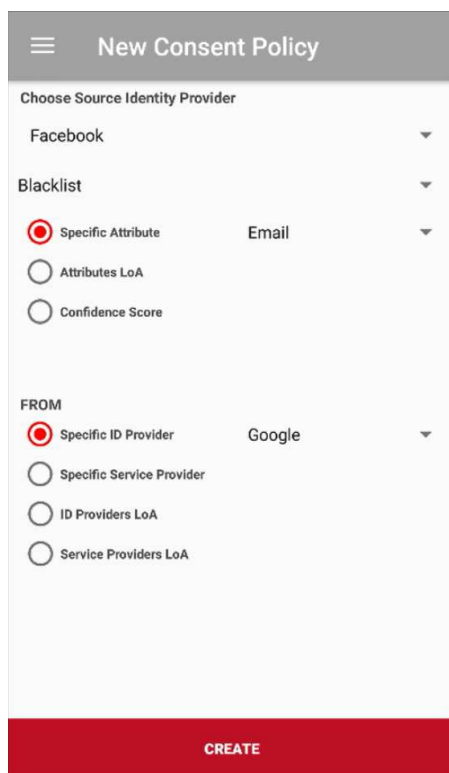


Figure 15 Create new Consent Policy

### 6.2.2 View and Manage Consent Policies

The user can see a list with all the consent policies that she has created. These attributes are grouped under whitelisting policies and blacklisting policies.

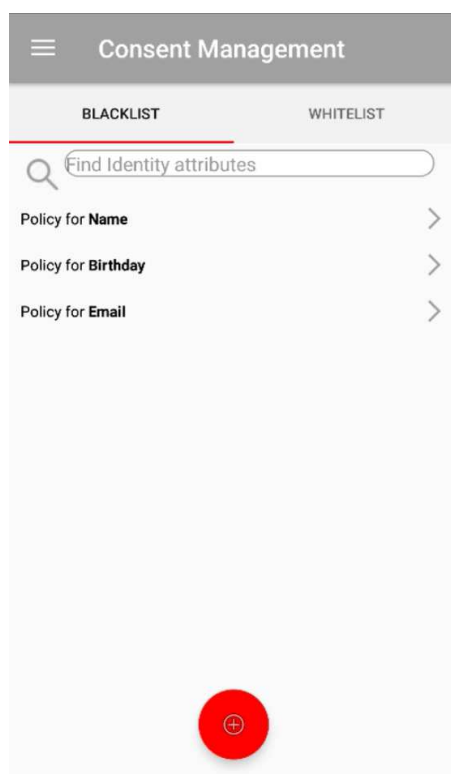


Figure 16:List of Consent Policies

Finally, the user can see more details regarding a consent policy by tapping on it (Figure 42**Error! Reference source not found.**). He can also long-tap on a policy, in order to delete it (Figure 43).

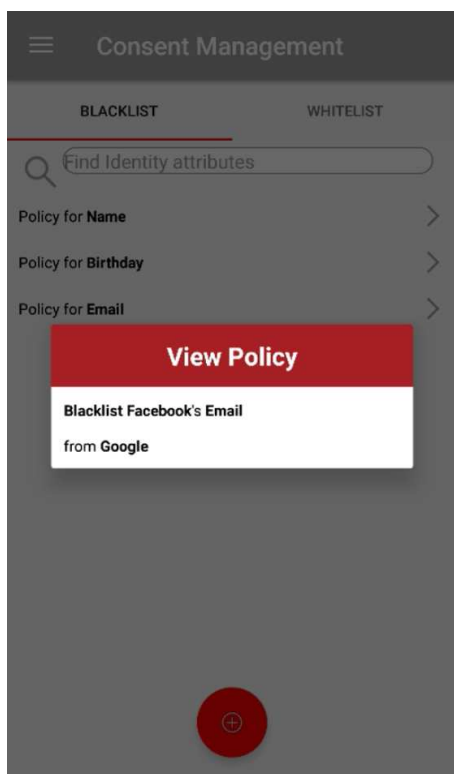


Figure 17: View policy details

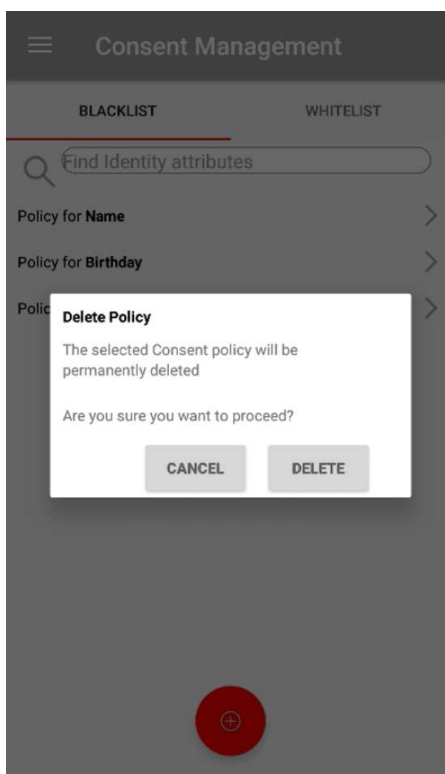


Figure 18: Delete a Consent Policy

### 6.3 Consent Management Web Interface

In this section we will describe and demonstrate the Web interface of the Consent Management module. At the main page of the Identity Consolidator, the user can navigate to the Consent Management Module by clicking the dedicated icon.

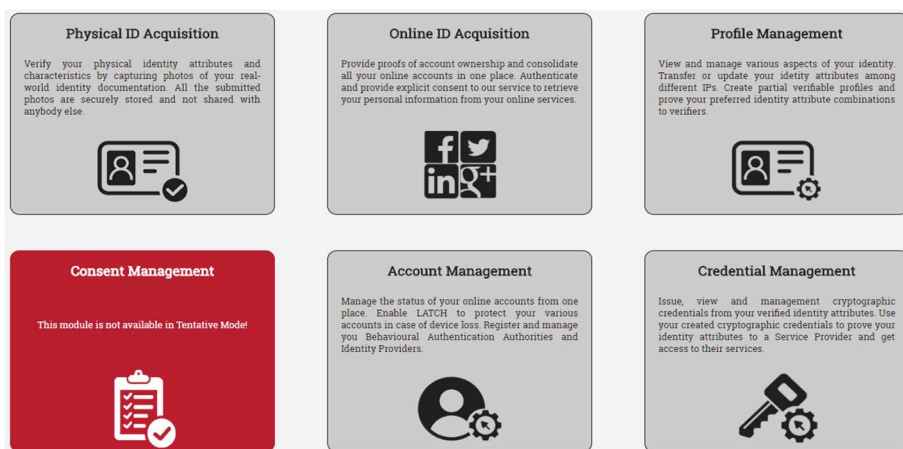
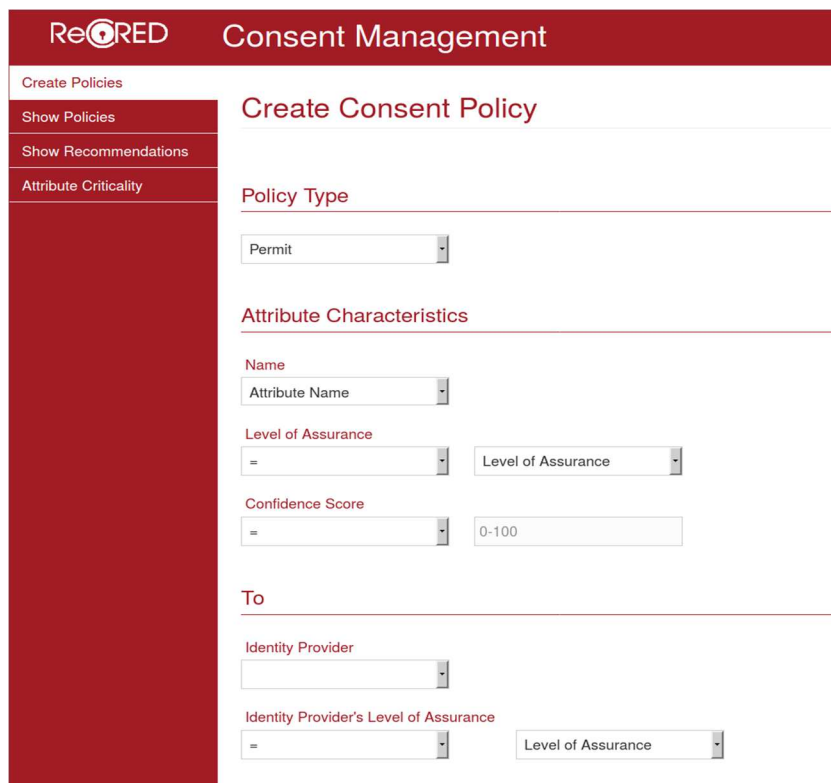


Figure 19. Consent Management Selection in IDC

### 6.3.1 Create Policy

In the Create Policies tab of the web interface, the user can define their consent policies. The policies are stored in xml under the XACML protocol and at the database for easier management. Specifically, a user is able to select the type of the policy (permit or deny), the source and destination IdP as well as the involved identity attributes. Essentially this functionality enables the user to set his preferences for the identity attributes transfer among Identity Providers.

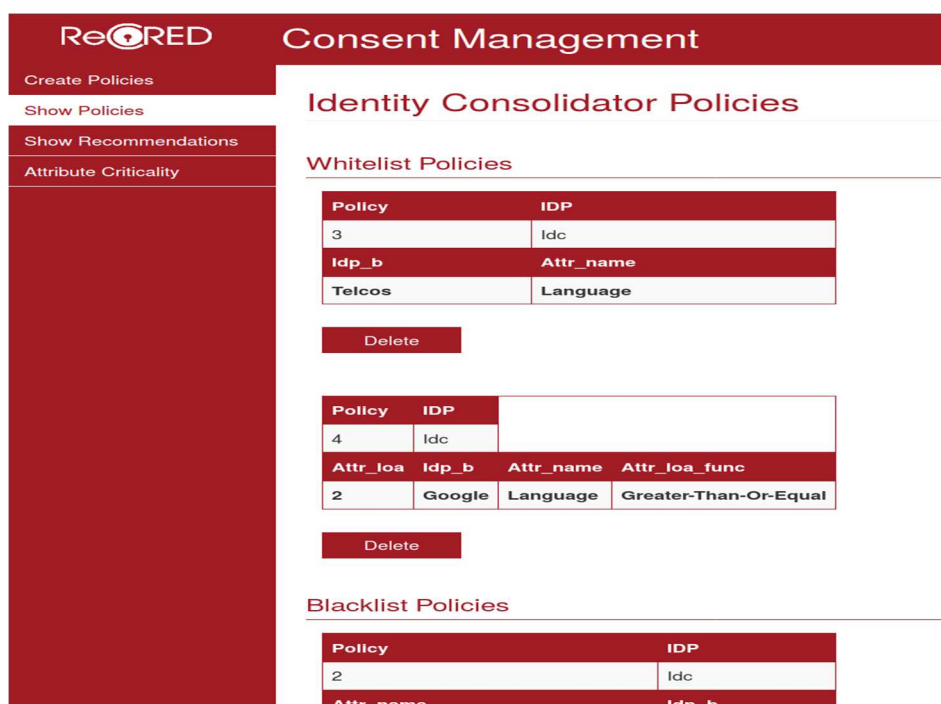


The screenshot shows the 'ReCRED Consent Management' web interface. On the left is a sidebar with navigation links: 'Create Policies' (highlighted), 'Show Policies', 'Show Recommendations', and 'Attribute Criticality'. The main content area is titled 'Create Consent Policy'. It contains several sections: 'Policy Type' with a dropdown menu set to 'Permit'; 'Attribute Characteristics' with fields for 'Name' (set to 'Attribute Name'), 'Level of Assurance' (set to '=') and 'Level of Assurance' (dropdown), 'Confidence Score' (set to '=') and '0-100'; and 'To' section with 'Identity Provider' (dropdown) and 'Identity Provider's Level of Assurance' (set to '=') and 'Level of Assurance' (dropdown).

Figure 20. CMM Policy Creation

### 6.3.2 View and Delete Consent Policies

After creating some consent policies, the user is able to view the already defined consent policies as well as deleting them. The figure below demonstrates the web interface for this functionality. Each consent policy can be deleted by clicking on the delete button.



**ReCRED** Consent Management

Create Policies  
Show Policies  
Show Recommendations  
Attribute Criticality

### Identity Consolidator Policies

#### Whitelist Policies

Policy	IDP
3	Idc
Idp_b	Attr_name
Telcos	Language

Delete

Policy	IDP		
4	Idc		
Attr_loa	Idp_b	Attr_name	Attr_loa_func
2	Google	Language	Greater-Than-Or-Equal

Delete

#### Blacklist Policies

Policy	IDP
2	Idc
Attr_name	Idp_b

Figure 21. CMM Policy View/Delete

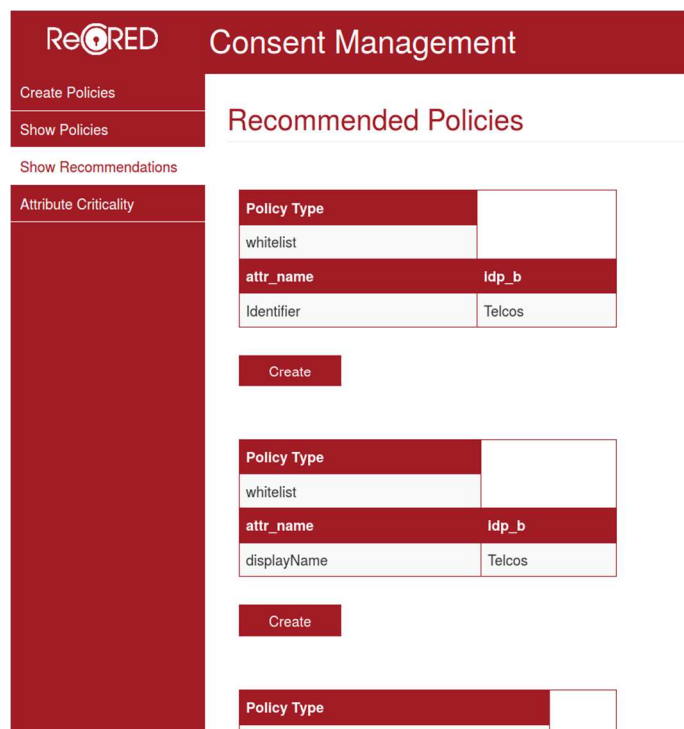
### 6.3.3 Consent Policies Recommendations

As the consent policies increase we want to assist the user in managing existing policies and recommend new ones. Rule based recommendations are simple and easy to derive in small scale systems. However, when the system gets more complex it becomes more difficult to find rules that will provide recommendations. Machine learning algorithms are used to derive the dependencies between information in an automate manner, can assist in deriving policy recommendations. In that regard, we utilize a machine learning model to provide policy recommendations to the user.

The model is based on a Markov Logic Network where each policy is assigned a weight of importance based on the active policies of all the users and the (transfer) requests logs. By assigning an importance weight on each policy we can provide recommendations to the user if the weight passes a specific threshold. This essentially means that as more users are active on the system the more accurate the recommendations are. Is worth mentioning that the threshold of each attribute is defined by the IDC's administrator. To limit the computational requirements of training the model, we resort to an offline process. That is, the consent policy recommendations and their associate weights are computed in weekly intervals or when enough requests are made that render the re-training essential.

Markov Logic Networks (MLN) combine Markov Networks with first-order logic; in our case the consent policies. In more detail, Markov Networks are undirected probabilistic graphical models that represent a joint probability distribution over a set of random variables; in our case the attributes. To soften the logic MLN associate a weight with each policy. Also, for the training procedure we require a finite set of constraints (domain) thus we transform the presence of any continuous attributes to discrete ones. This properly defines the domain and we train the model by maximizing the likelihood of the model given the data; in our case the request logs and the active policies. For recommendations, we search the whole domain space for other high probability policies and present them as recommendations.

At the Show Recommendations tab we present to the user consent recommendations based on the Markov Logic Network machine learning algorithm.

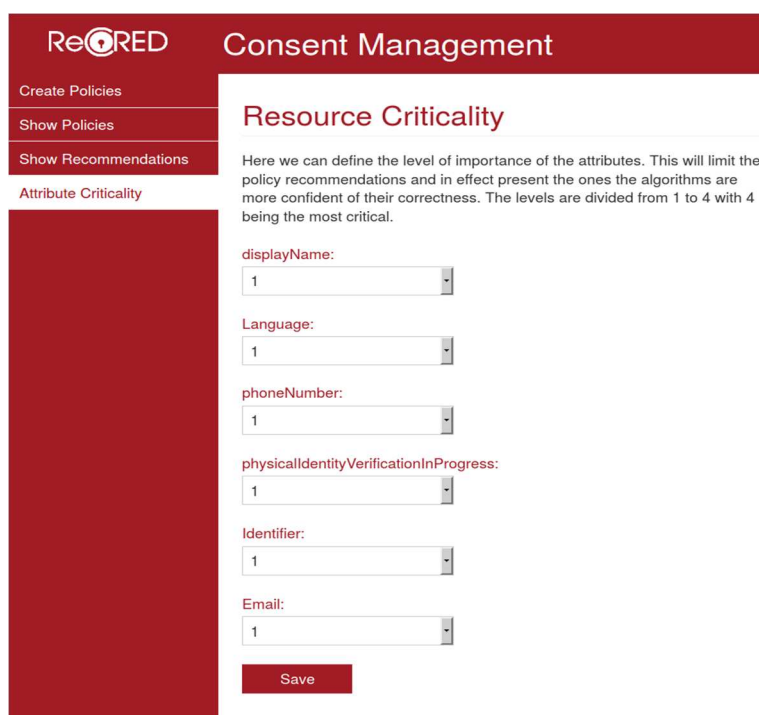


The screenshot shows the 'Consent Management' interface with a sidebar on the left containing three tabs: 'Create Policies', 'Show Policies', and 'Show Recommendations'. The 'Show Recommendations' tab is active. The main content area is titled 'Recommended Policies' and contains three policy recommendation forms. Each form has a 'Policy Type' dropdown set to 'whitelist', an 'attr\_name' field, and an 'ldp\_b' field. The first form has 'Identifier' in the attr\_name field and 'Telcos' in the ldp\_b field. The second form has 'displayName' in the attr\_name field and 'Telcos' in the ldp\_b field. Each form has a 'Create' button below it. The third form is partially visible at the bottom.

Figure 22. CMM Policy Recommendation

#### 6.3.4 Attribute Criticality

The identity consolidator administrator (only) can define the attribute importance. This will limit the policy recommendations for the critical resources. In essence, this increases the threshold on the policy weight (derived by the MLN) in order to be recommended.



The screenshot shows the ReCRED web interface. On the left is a sidebar with navigation links: 'Create Policies', 'Show Policies', 'Show Recommendations', and 'Attribute Criticality' (which is highlighted). The main content area is titled 'Consent Management' and contains a section for 'Resource Criticality'. This section includes an explanatory text: 'Here we can define the level of importance of the attributes. This will limit the policy recommendations and in effect present the ones the algorithms are more confident of their correctness. The levels are divided from 1 to 4 with 4 being the most critical.' Below this text are six dropdown menus, each with the value '1' selected. The attributes being configured are: 'displayName:', 'Language:', 'phoneNumber:', 'physicalIdentityVerificationInProgress:', 'Identifier:', and 'Email:'. At the bottom of the form is a red 'Save' button.

Figure 23. CMM Resource Criticality

## 7 Privacy Awareness

In this section we describe in more detail the privacy awareness tool for de-anonymization risk assessment that have been implemented in the context of ReCRED. This section is based on the description that exists in Section 3.3 of deliverable 4.3. We elected to also include this description on this deliverable too, as the basic primitives of the privacy awareness tool overlap with the primitives of WP5.

### 7.1 De-anonymization Risk Assessment

The IDC through the identity management will display to the user a risk figure indicating the possibility that an ID Provider or a verifier may infer the values of unknown user attributes based on the known user attributes that the ID Provider maintains for this user. The risk indicator is separate for each unknown attribute and ID Provider permutation.

This risk can be calculated only for attributes with known values, i.e. attributes whose values are stored in the Identity Repository. For attribute values stored only in ID Providers, it is impossible to determine their distribution and hence cannot calculate a risk factor.

Besides identity attributes view and management, the Identity Profile Management module offers the de-anonymization risk assessment functionality.

In general, de-anonymization risk assessment functionality helps a user to protect his privacy with respect to confidence in unauthorized identity attribute value inference. De-anonymization risk is calculated for identity attributes related to Identity Providers, Service Providers, and for all the



identity attributes that compose the financial information of a user. The de-anonymization risks that are taken into account involve risk of identity attribute value inference, and risk of user identification without his authorization.

Confidence in unauthorized identity attribute value inference is essentially the risk that an Identity Provider or Service Providers can infer the value of another identity attribute of a user that he has not shared with them based on the identity attribute values that the user shared with them and based on the distribution of the identity attributes across the web. In statistical terms it is an indication of where the user fits within the user population as this is segmented based on the revealed identity attribute values and the unrevealed identity attributes that the risk is calculated for.

The current implementation of this functionality addresses the risk of identity attribute value Inference for Identity Providers and for the financial information of the users.

Each one of the identity attributes where de-anonymization is applicable is associated with a confidence percentage that represents the possibility that an Identity Provider can infer the value of this attribute.

Calculation is based on the assumption that all the Identity Providers have acquired the distribution of identity attributes across the web from external means and this distribution is at least the same as the one that we have in the Identity Repository of the Identity Consolidator platform.

For example, assume that the ID Provider knows that out of 100 employed users, with income between 20000 and 30000, 30 have Overdue Loan Payments between 1000 and 2000. So, for an employed user, with income between 20000 and 30000, who has not revealed to the ID Provider that he has Overdue Loan Payments amount of 1500, the ID Provider can estimate that there is a 30% possibility to have Overdue Loan Payments amount between 1000 and 2000. Therefore, the de-anonymization risk for attribute Overdue Loan Payments amount of this user is 30%.

### 7.1.1 De-anonymization Risk Assessment categories

In general, the implementation of the de-anonymization risk calculation for ID Providers and Service providers is separated in two different categories depending on the method that the user is using to prove the (holding of identity attributes) and access the Service.

These categories are the following:

#### 7.1.1.1 Vanilla OpenID Connect

This is the case when a user uses the vanilla OpenID Connect to prove an identity attribute to a Service Provider. With OpenID Connect we have no untraceability and any Service Provider or Identity Provider can track the user across the web. In this case we want to protect the user against Service Providers and Identity Providers that has a unique identifier for each user and slowly by retrieving the identity attributes of each user they are in place to build a complete profile for each user. As a result, anywhere that a user goes to the web the Identity Providers that authenticates him are in place to trace him. The only thing we can do here is to prevent Identity Providers and Service Providers to build the complete profile of a user.

In order to achieve this, the Identity Profile Management module needs to be aware of the identity attributes shared with what Service Providers either from the Identity Consolidator or from an

external Identity Provider. Such information will be provided by OpenAM, which keeps logs each time a user shares an identity attribute with a Service Provider.

In general, the only anonymity that we can provide with vanilla OpenID Connect is that we can produce the confidence probability whether a Service Provider can infer the value of an attribute, which the user has not revealed to this Service Provider. So as soon as this functionality is implemented the user should be able anytime to go to the Profile Management module and see that will happen if he revealed a specific identity attribute or a combination of identity attributes. We should call this Privacy with respect to Confidence in unauthorized attribute value inference.

#### 7.1.1.2 Idemix and U-Prove

The second case is when a user uses ABAC (Idemix and U-Prove) to prove that he is an identity attribute and access a Service Provider. Using ABAC we have untraceability and unlinkability. With Idemix we have the Verifying Identity Providers who run an Idemix stack. In this case the user runs idemix with the verifying Identity Provider and with an idemix credential he proves to this Identity Provider one of his identity attributes or a combination of identity attributes (e.g., his age) and then the Identity Provider assures the Service Provider that the user is the holder of a credential that proves his age.

With Idemix we have two concepts of anonymity (untraceability and unlinkability - untraceable and unlinkable credentials). This is because when the same user shows a combination of three attributes to an Identity Provider and Service Provider at time A, if the same user goes to another Service Provider and prove the same three attributes at time B there is no mechanisms that can infer or assure that I am the same person as the one at time A. This means that “I am untraceable and my credentials cannot be linked”. This also means that no Identity Provider or Service Provider can build a complete profile for me.

Based on the aforementioned, in the case of Idemix the risk assessment is different and we need to calculate the risk per session. Additionally, with Idemix the calculation of the risk for de-anonymization does not depend on the credentials that a user shared before with Service Providers. Also, it does not depend on the policies that a Service Provider may have for access control but on the specific credential (identity attribute) or combination of credentials that a user is about to reveal to a Service Provider.

When we are talking about risk assessment with Idemix we need to know whether a given attribute can be inferred by a Service Provider and what is the confidence probability based on the combination of the credentials that a user is about to share with this Service Provider. For this we assume that all the Service Providers and Identity Providers have acquired the distribution of identity attributes and the distribution of the combinations of identity attributes (the frequency this combination of attributes occurs in the complete population) from external means.

For the computation of risk for de-anonymization in the case of Idemix we need to make sure, for each session separately, that when a user reveals some identity attributes (credentials) to the verifying Identity Provider or the Service Provider they do not learn any other of the user's identity attributes that he does not want to reveal to them. In general, we want whenever a user goes to an Identity Provider to inform him for example - given the distribution of identity attributes - that this combination of identity attributes that you are going to reveal also appear 10 times across the population and if you reveal them, the Service Provider has the knowledge to guess the values of some more of your identity attributes. This computation should be offered by the Identity Profile

Management module and the Idemix client that runs on the verifying Identity Provider should invoke the Identity Management module and display the calculated risk that the given Identity Provider or Service Provider can infer other identity attributes of the user.

## 8 TEE-based Secure Storage

An important functionality provided by ReCRED is the functionality of secure storage. As it is described in previous ReCRED deliverables, the user’s device manages multiple cryptographic credentials that were either sent by the ID Consolidator and ID Providers, or issued by the Idemix and U-Prove stacks deployed on the device. To maintain a high level of security, it is of great significance for these credentials to be securely stored, so they won’t be manipulated in case the mobile OS gets compromised. To this end, ReCRED takes advantage of the TEE functionality, as deployed in Android-based mobile devices, to develop and provide two important security mechanisms: a) *Cryptographic Credentials Storage (CSS)* to protect the cryptographic credentials and other important pieces of data (such as the facial templates enrolled using the Face Recognition module) and b) *Secure Key Storage (SKS)* to protect cryptographic keys, such as the Idemix user master secret and the U-Prove user private key.

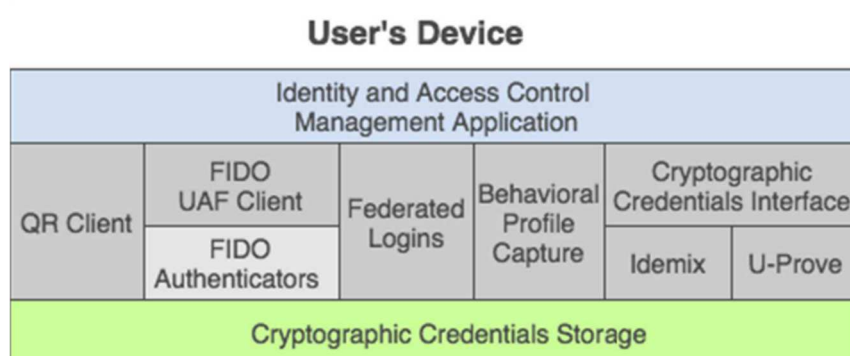


Figure 24 Cryptographic Credentials Storage on User's Device Protocol Stack

### 8.1 Cryptographic Credentials Storage

As it is documented in the Deliverable D5.1, Cryptographic Credentials Storage module was designed to leverage the Android TEE, if one is provided by the device, to securely handle the cryptographic credentials that were sent by the ID Consolidator and ID Providers, or issued by the Idemix and U-Prove stacks. In this deliverable, we describe all the enhancements made since the initial implementation.

In the initial phase of the module’s execution, an encryption key (256 bits) is generated (or restored) and is used along with the Advanced Encryption Standard (AES) algorithm. The key is bound to the specific algorithm and block mode, as well as to the purposes of encryption and decryption. The key is generated with the help of Android’s KeyGenerator class [1]. The encryption mode of AES that we chose to use is the Galois/Counter Mode (GCM) [3] which ensures the authenticity (integrity) and confidentiality of the encrypted data. Additionally, the ciphering is performed with the help of Android’s Cipher class [2] which provides access to cryptographic functions executed inside the TEE.

The Cryptographic Credential Storage (CCS) module was developed as a Java Class into an Android Library Archive (AAR). The Storage class implements the following public methods:

- `javax.crypto.SecretKey getCCSKey(String KeyName)`  
This function is able to create or restore an AES Key based on the given alias (KeyName argument) and returns the result as an object of the SecretKey API class.
- `byte[] encrypt(javax.crypto.SecretKey key, byte[] CryptoCredentials)`  
The encrypt function gets the generated or restored SecretKey object of the previous function call as argument and encrypts the plaintext argument (as byte array). The encrypted result is returned also as a byte array.
- `byte[] decrypt(javax.crypto.SecretKey key, byte[] encryptedData)`  
This function reverts the above process and returns the plaintext as a byte array.
- `byte[] secureStoreGet(String filename)`  
Based on the file name created in the previous method, this function retrieves the encrypted data into a byte array.
- `byte[] appendIvToEncryptedData(byte[] eData, byte[] iv)`  
This method performs the encryption procedure based on the processed data (eData) and the Initialization Vector (iv). It is also used by the `encrypt (SecretKey key, byte[] CryptoCredentials)` method in order to encrypt the plaintext input. The result is returned to the encrypt function as a byte array type.

The high-level overview of the Cryptographic Credential Storage (CCS) module is depicted in the figure below.

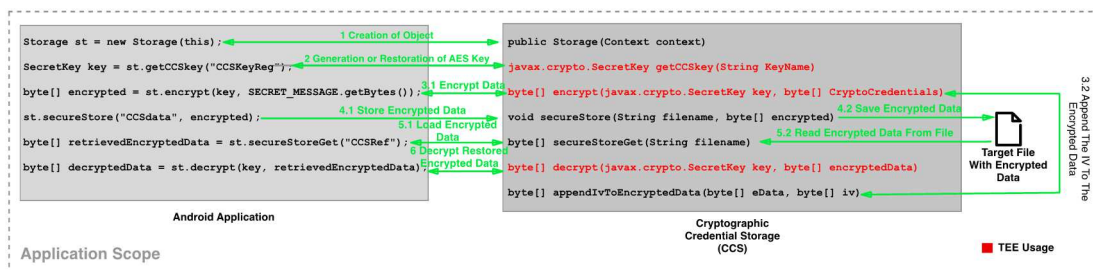


Figure 25 Cryptographic Credentials Storage overview

## 8.2 Secure Key Storage

By the time this document is written, the Android Keystore and the relative API [4] do not provide a way to store, encrypt/decrypt and handle conventional cryptographic keys using the TEE technology. These procedures, which are specifically handling cryptographic keys, are known in the literature as “Key Wrapping Algorithms” and they leverage symmetric encryption algorithms to encapsulate cryptographic key material. As it has been mentioned before, we are not able to run arbitrary code in the TEE in the form of a trusted application because these applications are pre-installed into the hardware by the hardware vendors and are bound to the device’s processor. For this reason, we employed the basic mode of the Advanced Encryption Standard Electronic Code Book (AES-ECB) to implement the specification of AES Key Wrap described in the RFC-3394 [4]. According to the specification, the length of the key to-be-wrapped must be a multiple of 64 bits [5].

We developed the Secure Key Storage module, able to generate an AES Key of 256 bits (for Electronic Code Book mode) that is going to be used to wrap and unwrap a given conventional cryptographic key using the Android’s API Keystore [6] and KeyGenerator [7] classes, with the help of TEE, if one is provided by the device. The secret cryptographic keys generated for the purposes of Idemix and U-Prove will be immediately wrapped by the device and will only be unwrapped when the user is authorized to use the application’s resources.

As we have already mentioned, the internal implementation of AES Key Wrap algorithm relies upon the AES Electronic Code Book (ECB) mode without padding. The encryption and decryption procedures behind the wrap and unwrap functions are performed with the help of Android’s Cipher class [8] which utilizes the TEE crypto implementation.

Secure Key Storage (SKS) was developed as an Android Library Archive (AAR). The SecureKeyStorage class implements the following public methods:

- `public SecureKeyStorage(android.content.Context context)`  
Initializes a new instance of the Class SecureKeyStorage.
- `public byte[] unwrap(byte[] wrappedKey, javax.crypto.SecretKey kek)`  
This method unwraps a key based on the RFC-3394 Standard.
- `public byte[] wrap(javax.crypto.SecretKey key, javax.crypto.SecretKey kek)`  
This method wraps a given key based on the RFC-3394 Standard.
- `public javax.crypto.SecretKey keyGen(java.lang.String alias)`  
This method generates (if it doesn't exist) or loads (if it exists) an AES 256-bit key based on the given alias.
- `public byte[] secureStoreGet(java.lang.String filename)`  
The method preserves the wrapped key into the directory:  
`/data/data/eu.recred.sks.secure_key_storage/files/[filename]`
- `public void secureStore(java.lang.String filename, byte[] wrapped)`  
The method preserves the wrapped key into the directory:  
`/data/data/eu.recred.sks.secure_key_storage/files/[filename]`

The high-level overview of the Secure Key Storage (SKS) module is depicted in the figure below.

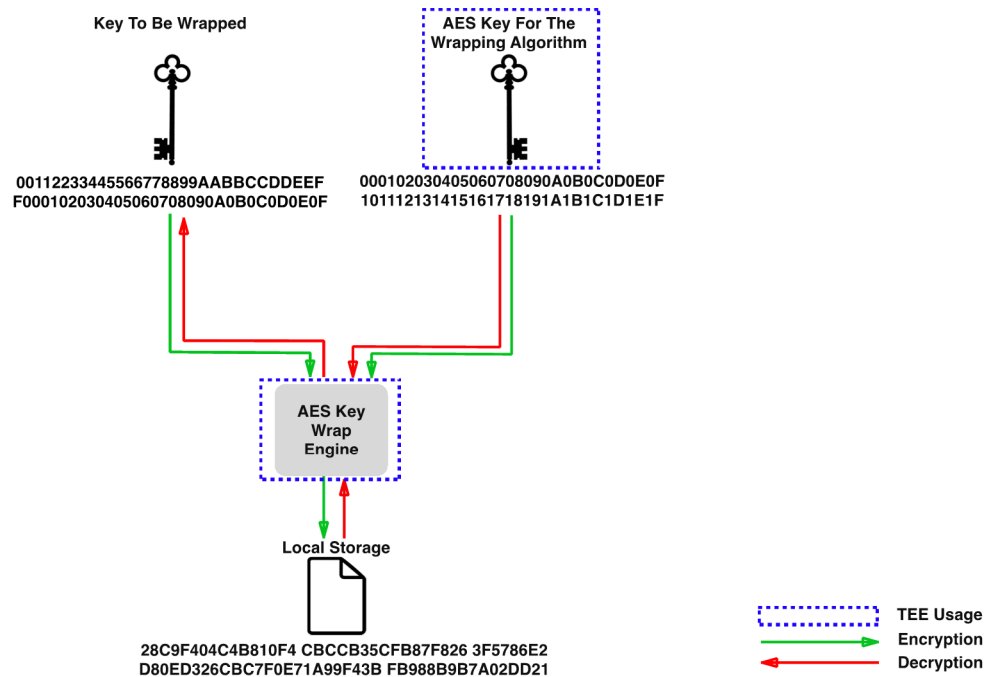


Figure 26 Secure Key Storage overview

## 9 Application Scenarios

### 9.1 Support to Financial Services

The financial services pilot focuses on the loan-origination use case. Bank clients requesting a banking product e.g. credit card or consumer loan, are required to present information including personal, professional, financial and other details to the banking institution, depending on which their request will be approved or rejected.

With the current infrastructure, anonymity is by default forfeited. Clients have to manually collect all the necessary documents and, on top of that, they reveal unnecessary personal details because the submitted documents include them by design. Finally, extra verification is necessary in order to ensure the authenticity of the submitted documents, and additional time will be spent profiling and scoring the client. This time-consuming and paperwork-intensive process can be vastly improved using the ABAC architecture.

Every piece of information that the banking institution requests can be individually revealed and certified by the corresponding authority (*Identity provider* in ABAC terminology), given the user's consent. The process can be automated since all data is electronically exchanged, and verification will be affected with the use of electronically signed credentials. Furthermore, clients requesting a loan may retain their anonymity if the banking institution decides that some non-revealing information from a reputable authority is sufficient to approve the client's request, for example instead of a full name and address the client presents only a Citizen's Identification Number from the Government Taxation Office.

### 9.1.1 Before ABAC

The current situation in the Greek Banking environment involves mostly manual procedures. The steps taken, starting from the request submission until the purchase of a banking product may differ depending on the product value and the risk involved. Extensive screening and detailed verification of personal and financial data is applied for high risk products such as business loans, mortgages etc. Simpler procedures are followed for lower risk products like consumer loans and credit cards. In all cases however, a lot of paperwork is required with reference to the documents that must be submitted with the client request, and a considerable amount of time will be spent for the verification of the submitted data.

Although credit cards request-forms can be submitted by post, for all other products the client will have to eventually visit the bank.

A bank employee will create a new (potential) client record and receive the client's request-form along with the necessary documents that list the client's personal information (financial, social security, health insurance, etc.). The documents' authenticity will have to be verified either by simple inspection or, depending on the value of the banking product, by direct contact of the originating authority.

In the case of collaterals such as real estate, external partners such as real estate professionals may be called upon by the bank to visit and inspect the property on location.

In order to complete the client's risk profile, the banking institution will collect additional data from organizations such as the National Banking Information System. This information concerns existing loans that the client has taken and loan payments that are overdue.

Finally, the banking institution may also collect information regarding the credit history of the client, and then calculate the risk of the client not being able to make the payments for the requested bank product.

At the end of the process the client's request will be approved or rejected based on the risk score and the customer's profile according to the banking institution's policy.

The “loan origination” use-case closely adheres to the “attribute based access control” paradigm, considering that the user's “banking profile details” are the attributes based on which the banking institution will control the user's access to the institution's services i.e. credit card issuance, consumer loan etc. The ABAC architecture provides considerable improvements regarding automation, efficiency, security, and privacy which are issues causing significant concern especially in the context of today's banking environment.

### 9.1.2 After ABAC

Once the ABAC architecture is adopted, it is expected that the certificate-issuing authorities as well as the banking institutions will deploy servers offering ABAC functionality.

#### 9.1.2.1 Identity Providers

The certificate-issuing organizations are the Identity Providers. They store and certify aspects of a user's identity such as full name, address, telephone number, social security status, health insurance status, annual income, financial obligations (outstanding balance), etc.

#### 9.1.2.2 Online Services

The banking institution is the Online Service offering its products to potential customers.



### 9.1.2.3 Behavioral Authentication Authorities

Organizations such as mobile telephony providers can function as Behavioral Authentication Authorities (BAA) providing second level authentication at the request of an Online Service. A mobile telephony provider may compare a user’s current location with the recorded user’s usual whereabouts based on the cell-towers that the user’s mobile device usually connects to. If there is no match the authentication will fail.

### 9.1.2.4 LATCH

Users may also configure BAAs to lock the account that a user has with an Identity Provider. In case multiple consecutive behavioral authentication attempts fail, a BAA’s ABAC server will use the LATCH service to lock all the user’s accounts according to the user’s specifications.

Taking all the above into consideration, the financial scenario is shaped as follows.

### 9.1.2.5 The Financial scenario revisited

A user that wishes to have a credit card or consumer loan issued by a banking institution will navigate to the banking institution’s web portal using the web browser of a mobile device (smartphone, tablet) or desktop computer. The bank’s web portal is the front-end to its ABAC infrastructure.

The user will be requested to provide certain information that will need to be validated. Such information includes, full name, age, annual income, social security status, employment status etc.

The user already has electronic credentials for some of these details stored in the mobile device. The user uses biometric authentication (fingerprint, voice, etc.) to unlock these credentials and submit them to the bank’s web portal.

The user also uses the mobile device to visit the web portals of other authorities and have electronic credentials issued for any additional information that the bank’s portal requests. The user either has a separate user-login with each of these authorities or may use OpenID and OAuth authentication in order to be identified at the authority’s server and have the necessary credentials issued. All the credentials are electronically signed by the issuing organization in order to ensure their authenticity. The credentials only include/reveal the specific information that the banking institution requests and not the total of the user’s records maintained by the credential-issuing authority. The issued credentials are transferred to the user’s mobile device and the user submits them to the bank’s web portal.

The bank’s ABAC server may in addition request for second level behavioral authentication. The BAA’s, e.g. Mobile Telephony Provider, ABAC server will retrieve the records of the user’s registered behavior, e.g. mobile cell-towers that the user mobile phone usually connects to, as an indication of the user’s normal whereabouts, and calculate the probability that the current user’s behavior matches the user’s profile or not. If the authentication fails, the BAA may also lock the user’s accounts at other ID Providers.

The bank’s ABAC server will examine and verify the credentials, consider the BAA’s response, and apply the banking institution “access control” policy. This policy specifies the criteria that must be fulfilled in order to accept or reject a client’s request for a credit card or consumer loan. If all criteria are met the user will be issued the credit card or granted the consumer loan.

## 9.2 Campus Wi-Fi and Campus-Restricted Web Services



The Campus Wi-Fi pilot focuses on the access of students, professors and guests to the Campus Wi-Fi and other Campus Web Services. The students and professors requesting access to one or more Campus resources (e.g. research network with the permission to print), are required to present information including personal (such as first and last name, age, street address, etc.) and educational (such as year of study, scholarship, teaching years, etc.) attributes. The guests must be vouched by a registered user in order to get access to some resources.

With the current infrastructure, the users reveal their full profile to get access to some resources with a lot of unnecessary information. They use the traditional username/password scheme to authenticate which nowadays is considered an insecure way. They also get access to various resources by simple access control policies.

The authentication and the authorization procedure can be vastly improved using the ABAC architecture. The pilot before and after ABAC and the advantages of using the ABAC architecture are presented below.

### 9.2.1 Before P-ABAC

When the students and professors register with their university, the university's IT services create and store their full profile within their infrastructure.

A user that wishes to get access to the campus Wi-Fi and the other resources can navigate to the University's web portal using the web browser of his/her mobile device.

Then the user uses his/her university username and password in order to authenticate to a particular web service against the university's Authentication (acts as the Identity Provider) and Authorization Server (acts as the Service Provider). This centralized component has a complete view of the user's profile and thus can provide the appropriate access to the resources.

The Authentication/Authorization Server maintains a role-based access control list. This has the disadvantage to not allow the definition of flexible fine-grained access control policies.

In summary, this paradigm has two important drawbacks; a user must reveal all of his identity to the Authentication/Authorization server and the network administrator is unable to define flexible fine-grained access control policies. This results in privacy concerns from the user's perspective.

### 9.2.2 After P-ABAC

When the students and professors register with their university, the university's IT services create and store some identity attributes (such as full name, title or department, etc.) within their infrastructure. Furthermore, privacy-preserving cryptographic credentials of the identity attributes are issued and stored on user's device by leveraging the developed storage that is based on trusted execution environment.

A user that wishes to get access to the campus Wi-Fi and the other resources (with some permission) can open the Campus Access mobile application and choose the network resources that he wants to get access to.

Then the Campus Access mobile application informs him/her for the identity attributes required for all the selected resources.

After the user's explicit consent to release the necessary cryptographic credentials from the device, the user authenticates with the Authentication Server (acts as the IdP) by using the released

cryptographic credentials. The Authentication Server, by leveraging the underlying cryptographic stacks, it authenticates the end-user and extracts the identity attributes from the credential.

After the successful authentication, the Authentication Server transfers the required identity attributes to the Authorization Server via the OpenID Connect specification. To achieve this, the Authentication Server is configured as an OpenID Connect Provider.

With the confirmed identity attributes, the Authorization Server checks against to its ABAC XACML-compliant access control policies and grants or denies access to those resources. The ABAC policies specify the criteria that must be fulfilled in order to accept or deny a user’s request. If all criteria are met the user can get access to the required resources.

#### 9.2.2.1 *Advantages of P-ABAC*

1. The users have the opportunity to use a very useful mobile application to get access to various resources instead of the navigation to the University’s web portal to find a resource.
2. The users do not have to validate their complete identity and their full profile in order to get access to a resource. This approach allows users to reveal only a set of required identity attributes and not to reveal irrelevant parts of their profile.
3. The Authorization Server grants access to the requested resources according to the complex attribute-based access control policies instead of the simple policies. The ABAC policies also take into account the level of assurance of the required identity attributes and the level of criticality of the requested resources.
4. The P-ABAC solution offers increased security and privacy for the end-users as the underlying cryptographic stacks offer the unlinkability and untraceability properties.

### 9.3 Age Verification

Age verification is based on UPCOM’s Age Gate product, an attribute-based solution which can verify if a user requesting access to an age-restricted online resource is above a certain age, without revealing any other personal data. The online resource could be an age-restricted web site (e.g. porn or violence related), specific content (e.g. an NC-17 movie) or a purchase (e.g. alcohol or tobacco). The providers of those resources do not need to know any personal information of the users, other than their age.

In addition, the Age Gate solution can also be used for age verification to physical places, such as nightclubs, casinos, purchasing alcohol from a liquor store, etc.

The Age Gate product uses various alternative methods in order to verify the user’s birthdate, such as trusted physical ID providers (government authorities, banks, universities, etc.) and electronic ID cards. It can then use ABAC in order to issue cryptographic credentials, including only the (verified) birthdate attribute, according to which the service provider can grant or deny access to specific resources, based on well-defined policies. Those credentials are issued to the user’s device, either by the ID consolidator or directly by a trusted IDP. The user can also backup those credentials to the IDC, so that they can be recovered, e.g. in case of device loss.

During the age verification application scenario, the following roles are identified:

- The User, who wants to prove his age, in order to be granted access to an affiliated online service that provides an age-restricted resource. For this, the User can issue a cryptographic credential on his date of birth, through an authorized Issuer, and securely store this credential in his mobile device.
- The Issuer, typically an IdP which has verified the user’s date of birth and can issue cryptographic credentials on his age. During the Age Verification pilot, the ReCRED Identity Consolidator acts as the issuing IdP.
- The Verifier, which is the Age Gate server acting as a verifying IdP. The User engages in a Show protocol, in order to convince the Verifier (Age Gate server) that he has the date of birth attribute.

Following is the course of actions for the anonymity-preserving Age Verification scenario:

1. The user attempts to visit an age-restricted website.
2. The website (acting as an SP) asks from the Age Gate Server (acting as an IdP) to verify the visitor’s age.
3. If the user visits the website using his PC or laptop, then the Age Gate server returns a QR code, which is displayed in the age-restricted website and the user can use his Age Gate mobile app, in order to scan the QR code and prove his age. Note that this step is not required if the user attempts to visit a website using his mobile device.
4. The Age Gate mobile app initiates user-to-device authentication using FIDO UAF.
5. Upon successful authentication, the Age Gate mobile app searches for cryptographic credentials in the user’s device, that prove the user’s age.
6. The Age Gate Server (acting as a verifying IdP) runs idemix / uprove, so it can verify the user's age (or whether the user's age is above or below a threshold) using the found credential.
7. The Age Gate Server evaluates the age policy defined for the requested website (e.g. age > 18), against the user’s actual age credential, and returns a true/false value to the website, using OpenID Connect.

If the user does not have a cryptographic credential on his age (or only has expired credentials), then she can issue new credentials through the Credential Management module of the ReCRED Identity Consolidator.

### 9.3.1 Before ABAC

Age verification is an extremely important, yet very difficult to solve problem, especially without sacrificing user privacy. Current approaches fall into three main categories:

1. The user has to demonstrate evidence of ownership of a document which proves that he is an adult, such as a credit card or a driver’s license. However, there are major issues with this approach. First of all, sensitive information is inevitably revealed and personal data could also be disclosed. In addition, this approach cannot guarantee that the user is the legitimate owner of the provided proof. For example, a child could use her parent’s credit card or driver’s license, in order to access age-restricted content. Last but not least, proper age

verification goes beyond the proof of being an adult. For example, the minimum age for drinking in USA is 21. Therefore, even if the user is the legitimate owner of a credit card, it does not necessarily mean that he is also above 21 years old.

2. The user has to demonstrate evidence of ownership of a document that explicitly states her birthdate, such as an ID card or a passport. With this approach, the service providers can determine the exact age of their users, however the rest of the problems with the first approach also apply here. Especially when it comes to the user's privacy, such legal documents usually include even more personal data than a credit card.
3. The service provider includes an age disclaimer, with which the user must agree in order to gain access to an age-restricted resource. This is a widely-used approach, which is mainly used as legal cover for the service providers, hardly preventing any minors from accessing age-restricted resources.

### 9.3.2 After ABAC

We strongly believe that ABAC, along with the acquisition of the user's physical identity, can offer a new approach to age verification, which provides a solution to all the problems related with the current approaches.

The exploitation of the ABAC architecture benefits all involved parties. More specifically:

- The users can access age-restricted resources without having to share with the service providers any personal or sensitive data other than the absolutely necessary (that they are above a certain age).
- The service providers can verify the age of their visitors and grant them access to age-restricted resources, according to specific policies that they can easily create and manage.
- Minors are protected against age-restricted content and/or products.

## 9.4 ISIC Student Discounts

The ISIC Student pilot is based on a product where ISIC operates a solution for students to buy items using a mobile application and get them in a physical store. The solution allows for targeted campaigns where end users can receive discounts based on attributes (e.g. university). Also, there can be constraints on other user attributes. For example, alcohol can only be allowed if a user is over 18.

Also in this pilot, the benefits of ABAC are obvious. Introducing the concepts of ReCRED architecture and PABAC allow eCommerce solutions to provide a better experience for both buyers and sellers. The solution will benefit on several aspects:

PABAC provides following advantages to ISIC:

- Because of the architecture, it is easier to get many identity providers (universities) to provide identities. Larger target groups mean more business value.

- Because ISIC only knows the absolute minimum of information, it makes it much easier to be compliant with privacy regulation (e.g. GDPR). This also minimizes costs to put other security controls in place.
- Information is more likely to come from trusted sources with verified and up to date data. This lowers the risk of fraud and wrong deliveries.

On the other side, PABAC provides privacy and a more convenient journey to end users.

- Only the bare minimum about users is known to the application: No one will ever know whether a user bought a certain product. Only if there are legal (being over 18) or commercial (you need to be a student) requirements; very specific generic information needs to be disclosed.
- Account details do not need to be kept up to date at all places: during each authentication, the solution will get the most up to date information. Before ABAC, when user data changed, users had to change the data in all systems. If a student moved to another dorm, the information needed to be changed in many systems. Using the ReCRED architecture in combination with PABAC, vendors can trust that any data received can be trusted and is up to date.

#### 9.4.1 Before and After ABAC

Bringing in ABAC in the ISIC Student Discounts accounts has come with many advantages.

- Before ABAC, there was only one identity provider. All users of the solution had to (manually) register with ISIC to be part of the solution. Using the ABAC infrastructure, the solution can be opened up to students of any university that can proof student status. This provides a much larger user base to the solution.
- Before ABAC, attributes were static and unreliable. When a user registered, information such as address was provided. However, when attributes change (e.g. if a student moved), many students did not seem to modify their address at ISIC. Chances of address changes being done are much higher at the university itself. As attributes are now retrieved from the university, data is more reliable.
- Before ABAC, there was close to no anonymity. Whilst vendors did not see any information about the user; ISIC could correlate all behaviour with the user's attributes. After ABAC, even ISIC can, if the user chooses, only see the things that are required to know that a user is allowed to get a discount or buy something.

## 9.5 Federated P-ABAC WiFi (WIFAB)

This section extends the *ABE-Based P-ABAC Solution for Wi-Fi* described (i.e. WI-FAB) in section 3.1.11 of D5.2 to the multi-authority scenario. At the time of this writing this work has been submitted to IEEE WiMob 2017 and accepted for publication in the conference proceedings.

User authentication at Wi-Fi Access Points (APs) is becoming an important issue. Wi-Fi APs are indeed ubiquitous, but existing authentication methods such as WPA/WPA2 static pre-shared secret key (PSK), or 802.1X-based online authentication services (e.g., RADIUS servers/proxies) have their theoretical or practical limitations. In a previous work, we proposed WI-FAB, a new authentication mechanism which neither requires online backend access control infrastructure, nor relies on a static pre-shared secret key. In this work, we extend WI-FAB by removing the need for having a central authority for user authentication and credential issuing. Our main contribution is twofold: (i) adopting *decentralized multi-authority CP-ABE*, we support the users who have authentication/authorization credentials from multiple authorities. We decouple the user credentials issuing from the management of the WPA2-PSK, so that neither the credential issuing authority can track the users, nor the AP can access the real identity of the users. Considering an extensive attack model, we show that the proposed approach is secure and preserves the privacy of the users. (ii) We provide a real-world implementation of the proposed approach on off-the-shelf embedded hardware to demonstrate its feasibility and efficiency.

### 9.5.1 Introduction

During the last decade, the number of Internet users has increased significantly, which, reported by International Telecommunication Union (ITU), was more than 46% of the world population by the end of 2016 [9]. This is due to the ever increasing number of mobile devices and connections, which will grow to *11.6 billion* by 2021, reported by Cisco [10]. However, due to the limitation of the existing cellular network (i.e., 3G, 4G, LTE), Cisco predicts the increase from 60% in 2016 to 63% by 2021 in offloading from the mobile data onto the fixed network through Wi-Fi or femtocell [10].

At the same time, providing a secure, privacy preserving, and straightforward method for authenticating the users willing to access a Wi-Fi network is challenging. This issue applies to both open and protected Wireless Local Area Networks (WLANs). In the former case, the user usually needs to register on a splash page in real-time, and after being authenticated he will receive the credentials to connect to the Internet. In such a scenario, service providers basically use an 802.1X-based authentication service leveraging a backend online infrastructure. This authentication method not only requires an online infrastructure, which is not always available, but also reveals sensitive information of the users, e.g., their identity or mobility patterns, to the credential issuing authority and the untrusted (or honest-but-curious) access points (APs) [11]. In case of protected WLANs, the user needs credentials, i.e., WPA/WPA2 pre-shared secret keys (PSKs) [12], to connect to the Wi-Fi. These credentials can be obtained offline or through another channel, which introduces several challenges (i.e., how to remember the password or how to keep it safe) and attacks [13], [14].

We believe that, in order to address the privacy and flexibility issues of traditional user authentication methods in WLAN, a desirable solution should satisfy the following requirements: 1) it should be easy for the admin of the network to configure the AP in such a way that the decision on who can access the Wi-Fi can be taken on the fly, by defining real-time access policies; and 2) it should be easy to refresh the WPA2-PSK in order to protect WLANs against unauthorized access (i.e., eliminating the need for updating all the users' credentials). In a previous work, we proposed *WI-FAB* [15], an attribute-based WLAN access control mechanism, without pre-shared keys and backend infrastructures. In WI-FAB, we satisfied the mentioned requirements by encrypting the WPA2-PSK that is used to secure the Wi-Fi connection adopting Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [8]. We enforced an access policy on the encrypted secret key based on the necessary attributes of the users who will be considered authorized to access the Wi-Fi. In WI-FAB [16], we left

as future work the scenario in which the defined policy on the encrypted PSK is written over attributes that are related to different authorities/domains. In particular, scenarios in which the users who would like to connect to the Wi-Fi AP are issued credentials from different authorities/domains. There are several real-world applications for such a situation, such as departments of a university that would like to take control of their students/staffs independently, city Wi-Fi, or a building equipped with a central Wi-Fi AP composed of several different independent companies/offices.

In this work, we adopt *decentralized multi-authority ABE* [17] and extend WI-FAB [15], while we inherit the main advantages of WI-FAB. We encrypt the WPA2-PSK specifying an access policy of attributes from multiple-authorities. We then divide the encrypted PSK into several chunks, insert each chunk in the WLAN beacons and broadcast them in the network. Upon receiving the beacons, a user who wishes to connect to the AP should merge the received chunks and reconstruct the encrypted PSK. If and only if a subset of the user’s attributes, which are associated to his secret key, satisfy the policy associated to the ciphertext, he would be able to decrypt and retrieve the PSK.

#### 9.5.1.1 Running example

Before introducing the key contributions of this work, we present a running example, which we will use throughout this section. Let us consider a university of  $n$  departments, e.g., Engineering, Economics, Medicine, History, and so on. Moreover, assume the university campus is equipped with a central Wi-Fi AP. In order to provide Wi-Fi access credentials for the users (i.e., students or staffs) in the campus, in a normal scenario there are two options. First, the users should receive a WPA2 secret from the central authority (CA) of the campus. Second, the campus can use an online backend infrastructure to authenticate the users. In such a scenario, users should receive credentials bound to their identity from the CA of the university. This way, i) users’ privacy is threatened, i.e., after every access to the Wi-Fi the credential issuing authority can track the users; and ii) there is a need for *online authentication* infrastructure.

This work provides the following main contributions:

1. Adopting *decentralized multi-authority CP-ABE* [17] we support users having authorization credentials from different authorities;
2. Considering an extensive attack model, we show that the proposed approach is secure and preserves the privacy of the users;
3. We provide a real-world implementation of our proposal on consumer-grade embedded hardware and demonstrate its feasibility through experimental results in a real testbed.

## 9.5.2 Background and Related Work

In this section we provide background knowledge on the concepts that we use in our proposed approach, along with a review of the most related work to our proposal in each subsection.

### 9.5.2.1 Attribute-Based Access Control and Encryption

Attribute-based access control (ABAC) [18], [19] is a flexible access control method in which the acceptance or rejection decision for accessing a data/resource is made based on the attributes of the requester. ABAC is indeed efficient in terms of communication overhead between the requester and the resource owner. This is due to the fact that the two parties do not need to agree on a pre-shared key to access the resource. A common tool that is usually used to provide ABAC is *Attribute-Based Encryption*, which we explain below.



#### 9.5.2.1.1 Attribute-Based Encryption

Attribute-Based Encryption (ABE) [20], is a public key encryption scheme, which enables the data owner to specify fine-grained access policies on the encrypted data. The access policy on the ciphertext is based on descriptive attributes (such as gender, or occupation). Two main forms of ABE are Key-Policy Attribute-Based Encryption (KP-ABE) [21], and Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [16]. In the KP-ABE scheme, private keys of the users are associated with the access policies, and a set of attributes is specified on the ciphertext. While in CP-ABE, the access policy is associated to the ciphertext, and private keys of the users are bound to a set of attributes that describes the user. A user is able to decrypt a ciphertext, if a subset of his attributes satisfies the access policy specified on the ciphertext. Due to the characteristics of ABE, several researchers have concentrated on ABE and proposed either new variations of ABE (to mention a few [17], [22], [23]), or new applications and performance evaluation of ABE (to mention a few [24]–[26]).

#### 9.5.2.1.2 Multi-Authority ABE

One important shortcoming of the first ABE schemes and most of their extensions is that the users' private key issuing is performed by a central trusted authority who should take care of authenticating the users and validating their attributes. Therefore, these schemes are not scalable for several different (distributed) domains in which the users might need to have different credentials reflecting their disjoint attributes (e.g., for their education, occupation, and so on). In order to address this issue, several researchers proposed *multi-authority ABE* to support private keys issued from different authorities having a hierarchical distribution, such as [27], [28]. In 2011, Lewko and Waters introduced a *decentralized multi-authority ABE* [17] scheme in which different authorities do not need to be aware of others, neither rely on a trusted central authority. Hence, there is no global coordination between the authorities other than a setup phase for generation of an initial set of global reference parameters (*GP*). This enables any party to serve as an authority and issue its public key along with a set of private keys for different users reflecting their attributes.

In our proposed approach, we take advantage of *multi-authority CP-ABE* [17]<sup>2</sup> in order to provide more flexibility in attribute selection and access policy definition, as well as removing the necessity of having a central trusted authority. In multi-authority CP-ABE [17] a data owner can define any access policy, composed of any chosen subset of attributes issued by any subset of authorities, over his encrypted data (refer to Figure 27, and for the notations refer to Figure 28). A user will be able to decrypt the encrypted data if and only if a subset of attributes associated to his private key, issued by any authority, satisfies the access policy on the ciphertext. As depicted in Figure 27 the users  $U_2$  (having attributes from authority  $A_2$ ) and  $U_3$  (having attributes from both authorities  $A_1$  and  $A_2$ ) are able to access and decrypt the data. While user  $U_1$  is not able to satisfy the policy and decrypt the data.

In the following, we explain five main algorithms of multi-authority CP-ABE as explained in [17]. Let us consider a set of authorities  $\mathbb{A} = \{\mathcal{A}_1, \mathcal{A}_2, \dots\}$  each of which having a pair of public key,  $PK_{\mathcal{A}_j}$ , and secret key,  $SK_{\mathcal{A}_j}$ , where  $j = \{1, 2, \dots\}$ . Hence, there is a set of public keys,  $S_{PK} = \{PK_{\mathcal{A}_j}\}_{j=\{1, 2, \dots\}}$ , in the system.

<sup>2</sup> While we are aware of the other more recent multi-authority CP-ABE schemes, such as [29], our main goal here is to show the effectiveness of using multi-authority ABE in anonymous user authentication in WLAN, no matter which of the existing multi-authority ABE schemes is used.



- **Global Setup.** Taking as input a security parameter ( $\lambda$ ); it outputs the global parameters  $GP$ ;
- **Authority Setup.** Taking as input the global parameters ( $GP$ ); it outputs a pair of public key  $PK_{\mathcal{A}_j}$ , and secret  $SK_{\mathcal{A}_j}$  for each authority  $\mathcal{A}_j$ . This algorithm is run by each authority, and in practice every authority generates one public/secret key pair for each attribute  $\gamma_i$  that it supports, i.e., it outputs  $\{PK_{\mathcal{A}_j, \gamma_i}\}$  and  $\{SK_{\mathcal{A}_j, \gamma_i}\}$ , while in Figure 27 we simplified this notation by having just one public/secret key pair for each authority;
- **KeyGen.** Taking as input the following parameters: the global parameters ( $GP$ ), a unique global identity ( $GID$ ), an attribute ( $\gamma_i$ , where  $i = \{1, 2, \dots\}$ ) which belongs to an authority ( $\mathcal{A}_j$ ), and the secret key of that authority ( $SK_{\mathcal{A}_j}$ ); it outputs a key  $K_{\gamma_i, GID}$  for a  $\langle GID, attribute \rangle$  pair associating the corresponding attribute to the specific identity (i.e., user);
- **Encryption.** Taking as input the following parameters: a message  $M$ , an access matrix  $(A, \pi)$ , the subset of public keys ( $\{S_{PK}\}_{i..j}$ ) of the relevant subset of authorities ( $\{\mathcal{A}\}_{i..j}$ ), and the global parameters ( $GP$ ); it outputs a ciphertext  $CT$ ;
- **Decryption.** Taking as input the ciphertext ( $CT$ ), the global parameters ( $GP$ ), and a subset of keys ( $\{K_{\gamma_i, GID}\}$ ) associated to  $\langle GID, attribute \rangle$  pair; it outputs the message  $M$  if and only if the set of attributes ( $\{\gamma_i\}$ ) associated to the keys “satisfies” the access matrix  $(A, \pi)$  defined on the ciphertext.

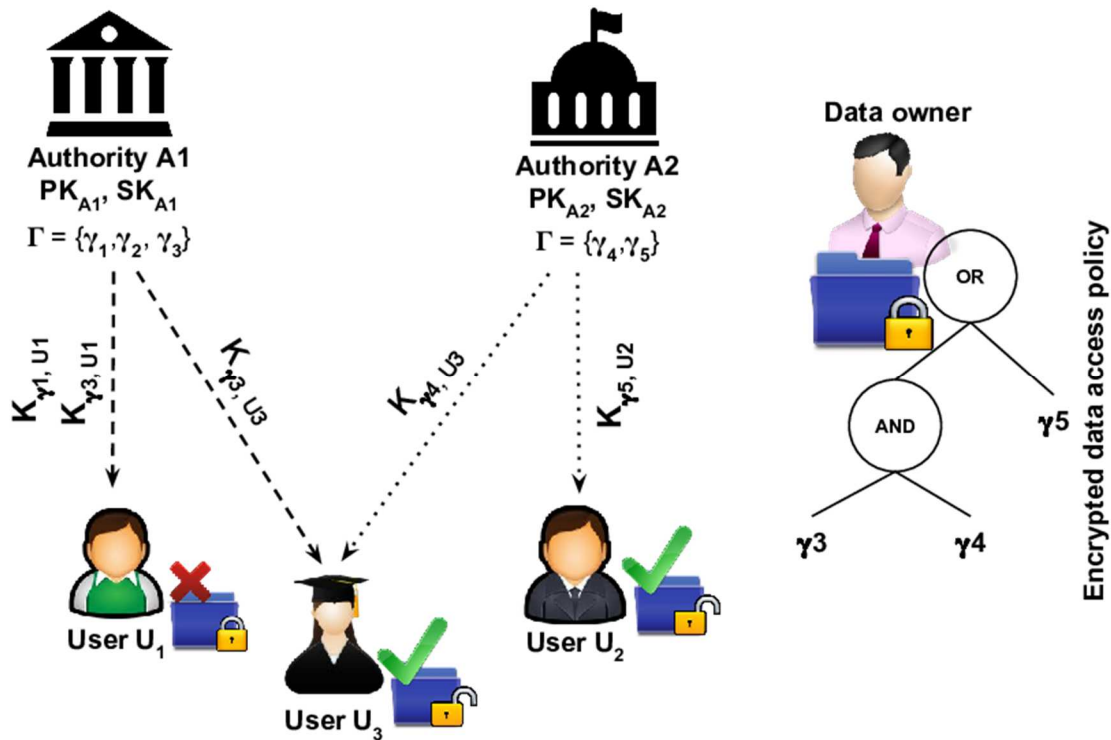


Figure 27 An example of Multi-Authority CP-ABE

#### 9.5.2.2 Wi-Fi Authentication and Access Control

The traditional Wi-Fi authentication and access control protocols, i.e., Wired Equivalent Privacy (WEP) and Wi-Fi Protected Access (WPA) are proved to be vulnerable against several security attacks [13], [14], which was the motivation for the introduction of the Wi-Fi Protected Access 2 (WPA2) protocol [12].

#### 9.5.2.2.1 WPA2 Protocol

The WPA2 protocol [12] is a rectification of the 802.11 standard that supports the use of Advanced Encryption Standard (AES). WPA2 guarantees data confidentiality and integrity for both personal and Enterprise authentication scenarios [30]. In the Enterprise mode, WPA2 uses IEEE 802.1X [34] in order to authenticate the users, while in the personal authentication scenario, users need to submit a PSK to be authenticated. It should be noted that, after the primary phase of user authentication, WPA2 creates a fresh unique session key for each connected user. This way, the secret key that is used for authenticating the user will be different from the key that will be used for further communication (i.e., message exchange) encryption.

#### 9.5.2.2.2 Anonymous User Authentication

Due to the importance of addressing the security and privacy issues of traditional Wi-Fi access control and authentications protocols (as explained in Section 9.5.1), several methods have been proposed to provide security, while privacy has gained less attention [14]. There are just a few proposals in the literature for providing (anonymous) privacy preserving user authentication. In [14], a private user authentication method based on Private Information Retrieval (PIR) is proposed, which preserves the privacy of the users against the AP and authentication server. Though the proposal is interesting, it still requires an online backend server to provide the users with the credentials for connecting to the AP, which increases user authentication time correspondingly.

Other fields of research related to our context are *anonymous* and *attribute-based* credentials [32], and related systems such as Idemix [33] and U-Prove [34]. However, neither of such schemes are proposed for WLAN access control, nor could be adopted in such a scenario, since they need a two-way communication (generally between a client and a server), while in our considered scenario we have a one-way communication from the AP to the user.

### 9.5.3 Models and Assumptions

In this subsection we explain our system model and the assumptions that we make in the remainder of the section, as well as the considered adversary model. We report the notations that we use throughout this section in Figure 28.

Notation	Description
$GID$	User’s global identifier
$GP$	Global parameters
PSK	WPA2 pre-shared secret key
$K_s$	WPA2 session key
$\mathbb{A} = \{\mathcal{A}_1, \mathcal{A}_2, \dots\}$	Set of authorities
$PK_{\mathcal{A}_j}, SK_{\mathcal{A}_j}$	Public and secret keys of authority $\mathcal{A}_j$ , respectively
$S_{PK} = \{PK_{\mathcal{A}_1}, PK_{\mathcal{A}_2}, \dots\}$	Set of public keys of the authorities
$\Gamma = \{\gamma_1, \gamma_2, \dots\}$	Universe of attributes
$U = \{U_1, U_2, \dots\}$	Set of users
$K_{\gamma_i, U_j}$	Secret key of the user $U_j$ ( $GID = U_j$ ) for attribute $\gamma_i$
$(A, \pi)$	Access matrix
$H()$	Hash function
$E_{MABE}$	Multi-authority CP-ABE encryption

$D_{MABE}$ 

Multi-authority CP-ABE decryption

Figure 28 Notation Table

### 9.5.3.1 System Model

In our model we consider a network consisting of the following entities: (1) A Wi-Fi access point (AP) which is protected with a WPA2 pre-shared secret key (PSK); (2) A set of users  $U = \{U_1, U_2, \dots\}$ , who intend to connect the Wi-Fi AP; (3) a set of authorities,  $\mathbb{A} = \{\mathcal{A}_1, \mathcal{A}_2, \dots\}$ , who provide the users with a secret key reflecting their attributes.

We assume that each user is holding a unique global identifier,  $GID$ , which could be the social security number of the user (consistent with [17], [27]). Each user  $U_j$ , for an attribute  $\gamma_i$ , receives a secret key  $K_{\gamma_i, U_j}$  associated to his  $\langle GID, attribute \rangle$  pair from the related authority. We also assume that each user can receive secret keys from multiple authorities for several attributes (see Figure 27).

Consistent with [17], in our system any entity (e.g., a building manager, a university or a single department in a university, a city municipality, etc) can become an authority and issue private keys to the users under its domain. We assume that there is no global coordination between these authorities, and they might not even know each other. Each authority can take care of several attributes, and in some special cases, several authorities can take care of the same attribute. Considering our running example (explained in Section 9.5.1), the “Student” attribute is the same in all the departments of a university, and so each department needs to assign secret keys for the “Student” attribute to its own students. In order to do so, similar to [9], we consider each attribute to be a string composed of: corresponding authority’s public key and the attribute name. For example, for the *engineering* department, e.g.,  $\mathcal{A}_1$ , and *medicine* department, e.g.,  $\mathcal{A}_2$ , the “Student” attribute will be *Student.PK<sub>A<sub>1</sub></sub>*, and *Student.PK<sub>A<sub>2</sub></sub>*, respectively.

We assume that distributed multi-authority CP-ABE algorithms [17] (explained in Section 9.5.2.1.2) are available in the system to be used by the authorities, AP and users.

### 9.5.3.2 Adversary Model

In this section we explain our considered adversary model, based on which we will provide security analysis in Section 10.1. In this work, we consider two types of attacker: a *passive attacker*, and an *active attacker*. A *passive attacker* can be one of the following entities:

- An *external eavesdropper*, who aims at accessing the Wi-Fi network by eavesdropping the channel and trying to obtain the PSK used to protect the Wi-Fi;
- An *honest but curious access point*, who honestly follows the protocol and provides the users with the encrypted version of the PSK, specifying a valid access policy over the attributes; while, it is curious in identifying the users and violating their privacy, e.g., mapping the users’ credentials to their identities;
- An *honest but curious credential issuing authority*, who honestly follows the protocol and provides the users with the secret key reflecting their attributes; while, it is curious in violating users’ privacy, e.g., tracking the users;
- An *internal eavesdropper*, who is a user successfully satisfied the access policy on the PSK and connected to the AP. This attacker aims at violating other users’ privacy by eavesdropping the traffic between a connected user and the AP.

An *active attacker* can be one of the following entities:

- *A set of colluding users on the attributes*, who do not satisfy the access policy enforced on the ciphertext and try to merge their attributes to obtain the PSK;
- *A set of colluding users on the PSK*, where one user in the set has successfully satisfied the access control on the ciphertext and obtained the PSK. This user aims to share the PSK with unauthorized users. This attack is actually a well-known attack, so-called *Alice and Bob Collusion attack (ABC)* [35];
- *An external DoS attacker*, who is able to perform two types of attack, both resulting in Denial of Service (DoS) to the users. The first attack is *expired beacon replay* attack. We consider this attack since in our proposed approach the AP can change the PSK periodically (or after each successful connection, or based on a specific event), and broadcast the new encrypted PSK, inside the beacons, to the users (the whole procedure is explained in Section 10.1). Hence, if an attacker replays the old expired beacons (i.e., the beacons that contain an old PSK), a user who receives and rebuilds such beacons cannot retrieve the new PSK and cannot access the AP. In the second attack, the attacker’s goal is to prevent the legitimate users from connecting to the Wi-Fi. This attack applies to the scenarios in which the AP refreshes the PSK after each successful user connection. Taking advantage of this feature, a malicious user whose attributes satisfy the access policy repeatedly decrypts the PSK and connects to the AP. By doing so, the attacker repeatedly triggers a PSK refresh command at the AP, which leads to preventing the other users to successfully receive the PSK and access the AP.
- *An external brute-force attacker*, who does not satisfy the access policy enforced on the ciphertext and performs a brute-force attack to obtain the PSK;
- *A revoked user*, whose attribute-based secret key is revoked, but she is trying to decrypt the PSK and access the Wi-Fi. The revocation of the key could be due to the expiration of his attributes (e.g., a graduate student is not supposed to have a secret key for the “Student” attribute) or misbehavior of the user (e.g., the user leaked the attribute’s key).

#### 9.5.4 Proposed Approach

We now present our multi-authority attribute-based access control approach for WLANs in two parts: (1) the AP side procedure, and (2) the user side procedure. In Figure 29, we provide an example scenario of our proposal. In this figure, we consider three users (borrowed from Figure 27) who intend to access the Wi-Fi AP, each of which is assigned attribute-based secret keys from one or more authorities as explained in Section 9.5.2.

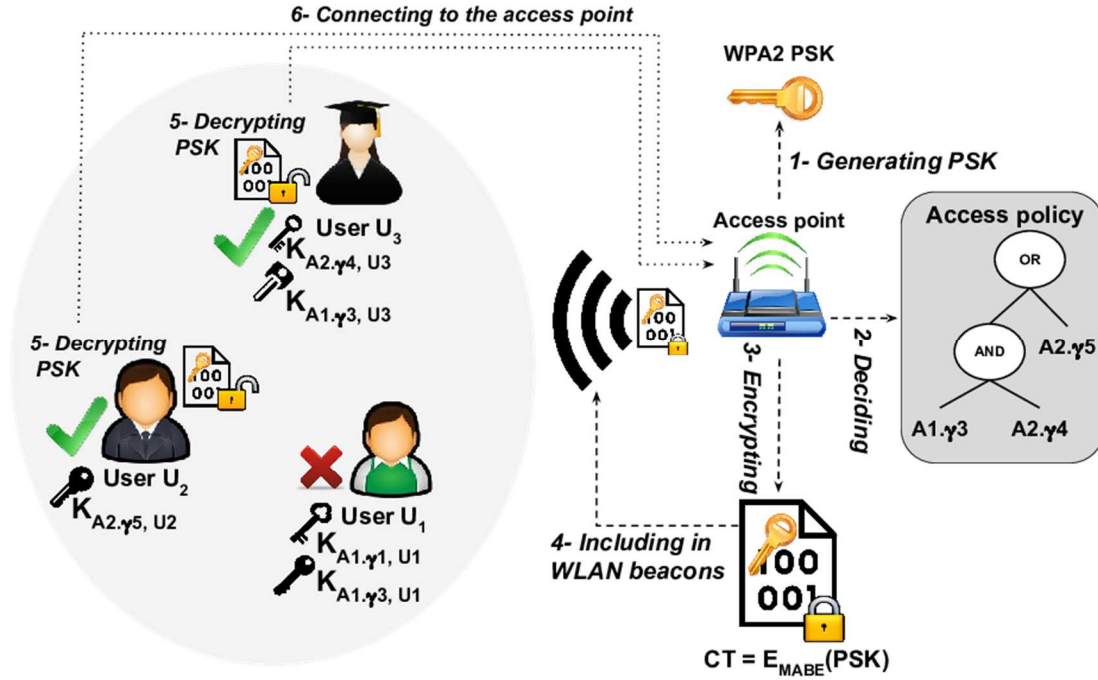


Figure 29 An example scenario of the proposed approach

- *The AP side:* the Wi-Fi AP generates a random WPA2-PSK that protects access to Wi-Fi (Step 1 in Figure 29). Then, it generates an access matrix  $(A, \pi)$  based on which defines the access policy  $\pi$  over the users' attributes (Step 2 in Figure 29). In the next step, the AP runs multi-authority CP-ABE *Encryption* algorithm [17] (explained in Section 9.5.2.1.2) and takes as input the PSK, the access matrix  $(A, \pi)$  over specific attributes, the public keys of the relevant authorities (related to the considered attributes), and the global parameters. The AP encrypts the PSK and outputs  $CT = E_{MABE}(PSK)$  (Step 3 in Figure 29). Then, the AP embeds the  $CT$  in the IEEE 802.11 information elements of the beacons (for information about the beacon management frame please refer to [36]) and sends to the users in its range (Step 4 in Figure 29).
- *The user side:* a user  $U_i$  who intends to access the WLAN captures beacons (by scanning or sniffing) from the access point  $AP$ . Extracts the IEEE 802.11 information elements containing  $CT$ . Checks if a subset of the secret keys associated to his attributes satisfies the policy  $\pi$  on the  $CT$ . If yes, he runs multi-authority CP-ABE *Decryption* algorithm [17] (explained in Section 9.5.2.1.2), i.e.,  $PSK = D_{MABE}(CT, GP, \{K_{\gamma_i, U_i}\})$  and obtains PSK (Step 5 in Figure 29). In the example scenario in Figure 29, only  $U_2$  and  $U_3$  are able to satisfy the policy. Subsequently, the user connects to the AP using the obtained PSK (Step 6 in Figure 29). After being successfully authenticated, the user and the AP negotiate on a unique fresh session key  $K_s$  which will be used for further communication encryption.

We propose the usage of PSK by the WPA2 protocol as a per-client pre-shared key which can be used only one time by a user, however PSK refreshing could be done also periodically or event-based. As soon as *one* user successfully connects to the AP, the AP performs the above-mentioned procedure (1) for the next user, generating and distributing a *new randomly generated* PSK. It is worth mentioning that, an important feature of our proposal is that, it enables the AP to refresh and randomize the PSK periodically, or per connected user. While, in the traditional Wi-Fi access control mechanisms, the pre-shared secret key refreshing is a challenge, as explained in Section 9.5.1. Note

that, due to the features of the WPA2 protocol, when the AP generates a new PSK, the users which are already connected to that access point will still stay connected. This is due to the fact that PSK is only used for authentication, while further communication between the user and the AP is secured with an agreed session key  $K_S$ .

In order to transmit the  $CT$  to the users, similar to WI-FAB [15], we can use fountain coding (please refer to [37]) to encode the  $CT$  in the IEEE 802.11 information elements. Information elements allow a maximum payload size of 255 bytes, but the  $CT$  can easily exceed this limit. To address this issue, we divide the  $CT$  into smaller chunks, encode it into *droplets* through fountain coding, and broadcast them. The users who are willing to obtain the  $CT$ , need to capture enough of these droplets and use fountain (de)coding to integrate these droplets and reconstruct the original  $CT$ . After this step, the user is able to use the multi-authority CP-ABE decryption algorithm (if his attributes satisfy the access policy) and recover the PSK.

### 9.5.5 Implementation and Results

Our implementation is targeted, on the access point side, at running on low-end routers equipped with operating systems for embedded devices such as OpenWrt (a Linux-based distribution for routers) or LEDE (an active fork of OpenWrt) [39].

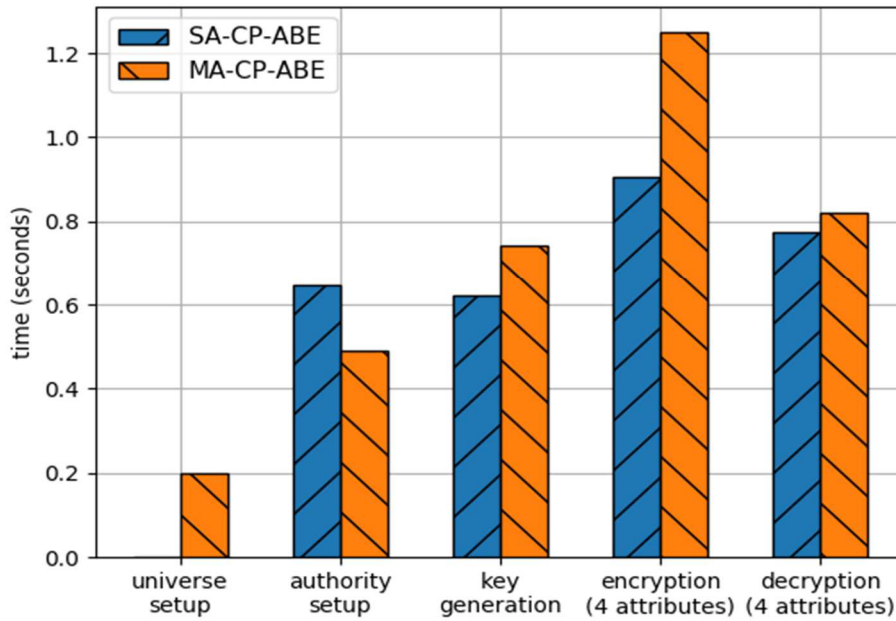
One of the main contributions of this work is the successful port of the Python Charm framework [40] (together with the GMP and PBC libraries) to the MIPS architecture. We provide openly available package feeds for the firmware build systems of LEDE and OpenWrt<sup>3</sup>. Our implementation of the proposed approach did not require modifications to the OS kernel: the current Linux mac80211 architecture already exposes the needed APIs (nl80211) to the user space. These APIs are employed on Linux-based operating systems by tools such as *wpa\_supplicant* (station side) and *hostapd* (access point side). We instead changed slightly (a 1-line patch) the *wpad-mini* daemon (a stripped down version of *hostapd* [41], included by default in LEDE and OpenWrt) to avoid disconnecting the stations when reloading the configuration file. Indeed, *wpad-mini* can be instructed, through a HUP signal, to re-read at run-time its configuration file. Moreover, the configuration of *wpad-mini* may include a parameter named `vendor_elements` which allows to specify, through a hexadecimal string, the content of the vendor specific information elements of the IEEE 802.11 beacons [36]. We actively make use of these features of *wpad-mini* to update the information in the beacons at run-time. To this aim we have developed a Python user-space daemon which runs also on the access point and controls the operation of *wpad-mini*. This daemon requires the CP-ABE public parameters and an attribute-based access policy for the WLAN in order to perform the following operations: (i) generates a random WPA2 secret, encrypts it with CP-ABE using the provided policy and serializes the ciphertext. This operation is performed both for single-authority CP-ABE and multi-authority CP-ABE using the Charm framework; (ii) divides the ciphertext into droplets (see Section 9.5.4); (iii) cycling through the chunks: the chunk is converted into a hexadecimal string. This string is used to update the `vendor_elements` parameter of the *wpad-mini* configuration file. A HUP signal is sent to the running *wpad-mini* to instruct it to re-read its configuration; (iv) depending on the settings, the above steps are repeated either when a station associates to the access point or at regular intervals.

<sup>3</sup> <https://github.com/netgroup/wifab-openwrt>



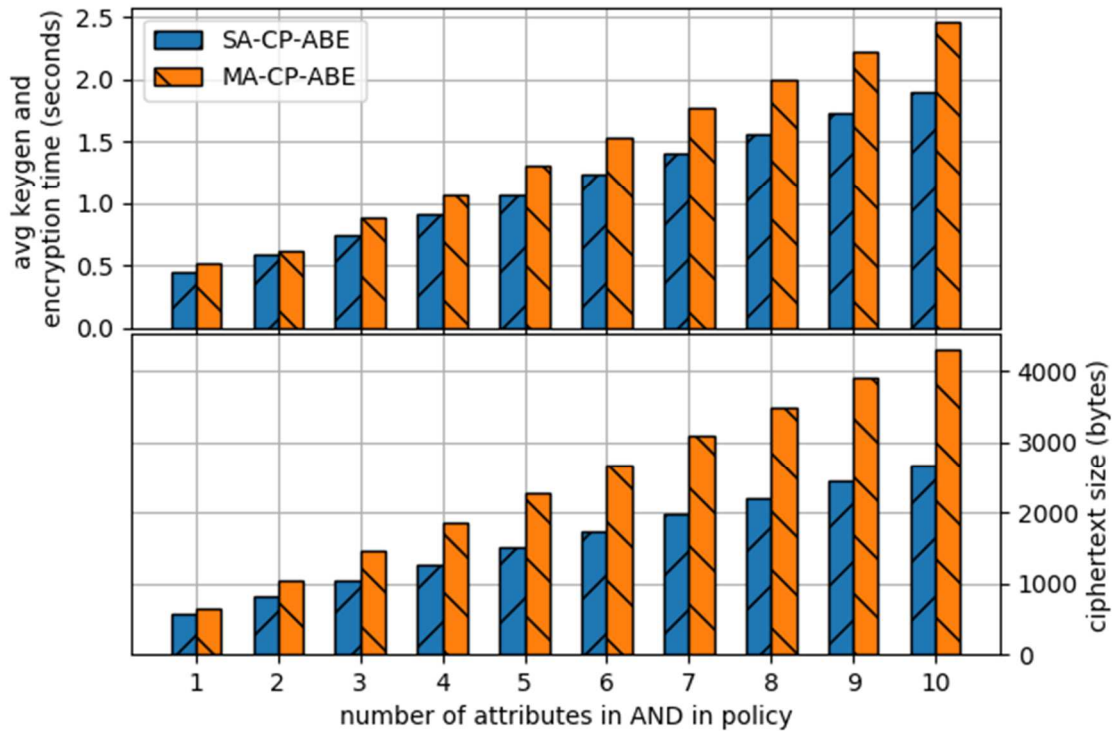
### 9.5.5.1 Experimental Results

To test the feasibility of our approach in a real-world scenario, we employ off-the-shelf consumer-grade hardware. Specifically, we use a Netgear R6100 router (Atheros AR9344 SoC, 560 MHz CPU, 128 MB RAM, 128 MB Flash) as access point, while Lenovo T450S laptops are employed as clients/stations. On the router we install LEDE and the software components as described above.



**Figure 30** Average times of main cryptographic operations executed on the router (a low-end MIPS device) using the ported Charm framework

The performance of the main single-authority CP-ABE [16] (here referred as SA-CP-ABE) and multi-authority CP-ABE [17] (here referred as MA-CP-ABE) operations executed on the router are summarized in Figure 30. The depicted times are averaged over 30 runs. Please note that SA-CP-ABE does not include a universe setup operation. Our results show that a low-end embedded device with a slow CPU can perform most of the considered operations in at most a few seconds.



**Figure 31 Average key generation and encryption times on the router vs. number of ANDed attributes in policy and ciphertext size vs. number of ANDed attributes in policy**

For our scenario we are especially interested in the performance of the WPA2 key generation and encryption, as these operations are performed on the access point. Figure 31 shows the average key generation and encryption times as well as the size of the generated ciphertext vs. the number of attributes in the policy. A random key is generated and then encrypted using policies with an increasing number  $i$  of attributes (from 1 to 10). The policies that we are using for the experiments are in the form  $\langle \gamma_1 \text{ AND } \gamma_2 \text{ AND } \dots \text{ AND } \gamma_i \rangle$ , as this form provides the worst case scenario from the computational point of view. The results are averaged over 30 runs. Please note that as the information elements may contain a limited payload (255 bytes), the size of the ciphertext affects the number of chunks/droplets that are required to reconstruct the secret.



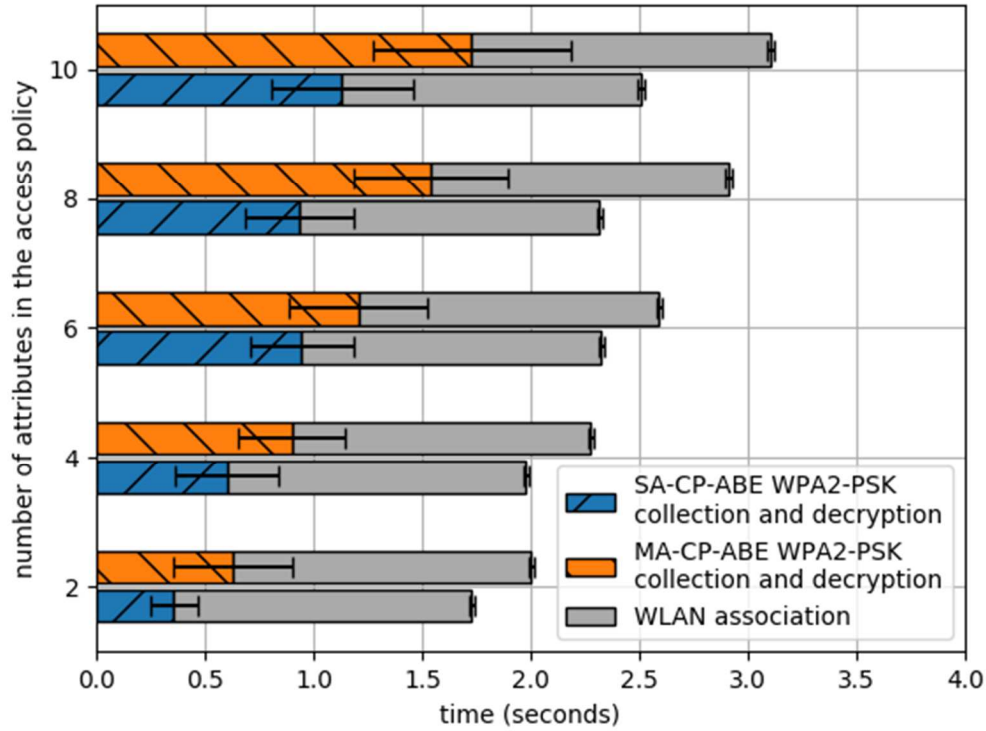


Figure 32 Required client connection times vs. number of attributes

Figure 32 illustrates the average time (over 25 runs) needed by the clients/stations to connect to the access point using the proposed approach. In the first phase, the chunks included in the beacons are collected to reconstruct the ciphertext and the ciphertext is decrypted. Then the decrypted WPA2-PSK is used to perform the standard WPA2 WLAN association. As it can be seen in the figure, the overhead added to the WLAN association times is comparable with the average time used to connect to WPA2-protected access points and usually accepted by Wi-Fi users. In other words, our proposal does not impose too much delay for connecting to the AP, compared to the traditional method where our proposal is not in place.

### 9.5.6 Discussion and Conclusion

In this section we proposed a multi-authority attribute-based access control mechanism for WLANs. In the proposed approach, which is an extension of our previous work WI-FAB [15], adopting *decentralized multi-authority CP-ABE* [17], we facilitate the access control for scenarios in which the users of the WLAN are issued attribute-based credentials from multiple domains/authorities. The advantages of the proposed approach are: (i) authentication of the users neither relies on a central authority, nor on an online backend infrastructure; (ii) it preserves the privacy of the users by authenticating them based on their profile attributes, not their identity; (iii) it addresses the PSK refreshing issue of the traditional WLAN access control systems. Considering an extensive attack model, we discussed and proved the security of our proposal. Moreover, our real-world experiments demonstrate the feasibility of our approach using off-the-shelf low-end embedded hardware, without requiring major (i.e., kernel-space) firmware modifications. Our results show that the overhead added by the chunk collection and key decryption phase is of the same magnitude of the time required for standard WLAN association.

The main challenges of our proposed scheme, which we leave as future work are: 1) the scalability of the system, and 2) the revocation of the users. The challenge of scalability comes from the fact that the same attribute (e.g., “student”) issued by different authorities (e.g.,  $\mathcal{A}_1$  and  $\mathcal{A}_2$ ) is treated as different attributes (i.e.,  $\mathcal{A}_1$ .student,  $\mathcal{A}_2$ .student), which is not desirable in large scale scenarios. Actually, we anticipate that this issue could be mitigated by supporting the aggregation of such attributes. Considering the user revocation challenge, different from other use cases of ABE (e.g., access control on the encrypted data, in which the data owner does not have control over the encrypted data after publishing it), in WLAN scenario a backend infrastructure is in place. Therefore, a possible promising solution for user revocation could be two-phase authorization system: in the first phase, adopting our proposed approach, the user who satisfies the access policy connects to the Wi-Fi infrastructure. In the second phase, the user needs to prove that his privileges have not been revoked, for which we could adopt Certificate Revocation List (CRL) in the system, along with anonymous credential systems such as *Idemix* [33] or *U-Prove* [34].

## 10 Privacy and Security Considerations

### 10.1 Federated P-ABAC WiFi

In this section, referring to the adversary model we considered in Section 9.5.3.2, we discuss the security of our proposed scheme against each of the considered adversaries.

#### 10.1.1 Passive attacker

Considered passive attackers are:

- *External eavesdropper*: the proposed scheme is secure against such an attacker, since if the attacker is not able to satisfy the access policy specified on the encrypted PSK, she is not able to decrypt and obtain the PSK. The security of the scheme relies on the security of the multi-authority CP-ABE against polynomial time attackers, for the proof please refer to [17].
- *Honest but curious AP*: the proposed approach does not reveal any sensitive information about the users, i.e., users’ credentials and their identities. The only information about a user that could be accessed by the AP is a set of user’s attributes that satisfies the access policy. However, this set of attributes highly depends on the combination of attributes defined in the access policy. For example, if the access policy is in the form of  $\langle \gamma_1 \text{ OR } \gamma_2 \rangle$ , then the user who has satisfied the access policy might have  $\gamma_1$  or  $\gamma_2$  with the same probability. However, if the access policy is in the form of  $\langle \gamma_1 \text{ AND } \gamma_2 \rangle$ , then the user for sure has both  $\gamma_1$  and  $\gamma_2$  attributes. Another point here is that mapping between the attributes and the users’ identities, is not trivial. Since the user never reveals his identity, and there is no clear bound between the users’ identity and attributes, the AP is neither able to link the attributes to the users’ identity, nor identify the users.
- *Honest but curious credential issuing authority*: the proposed approach preserves the privacy of the users against such an attacker, since the AP authorizes the users offline and the issuing authority is not involved in the user authorization procedure at the AP.
- *Internal eavesdropper*: the proposed approach is secure against such an attacker and does not violate the privacy of other connected users. This is due to the fact that each user who can satisfy the access policy is only able to retrieve the PSK key which is used to connect to the AP. While further communication between the AP and the users is encrypted using a session key,  $K_s$ , which is unique per user.

### 10.1.2 Active attacker

In the following, we discuss the security of our proposal against considered active attackers:

- *Set of colluding users on the attributes*: relying on the multi-authority CP-ABE scheme [17], our proposed approach is secure against collusion attack. This is due to the usage of the users' global identity ( $GID$ ), which binds the user's attributes to that specific user. In particular, in multi-authority CP-ABE, the encryptor blinds the message with some shares of a secret, and the decryptor can recover the blinding factor if and only if a set of keys associated to his  $\langle GID, attribute \rangle$  pair satisfies the policy. Since two colluding users will have different  $GID$ , they can neither recover the blinding factor, nor decrypt the message. For more details, please refer to [17].
- *Set of colluding users on the PSK*: this attack does not only relate to our protocol or Wi-Fi access control, but it also applies to any other kind of password-based access control on any data/resource. In particular, in any secured system with a password, there could be a user who successfully passes the authentication checks, obtains the key, and shares the key with other colluding users. Our approach is not safe against such an attacker.
- *External DoS attacker*: in our proposed scheme, similar to all the other Wi-Fi access control schemes, DoS on the users is unavoidable. Considering both attack scenarios (i.e., message replay and key refresh triggering attacks), refreshing the PSK could be done either after every single connection or periodically (even due to a specific event). Actually, in our proposed method, the admin of the AP can perform a trade-off between the connection delay and security by adjusting the suitable time to refresh the PSK. Moreover, going back to the second attacker (i.e., in the key refresh triggering attack), all the users (legitimate or attacker) have more or less the same probability of getting the beacons and decrypting the key.
- *An external brute-force attacker*: in order to protect the system against such an attacker, we propose to generate a random key, of the maximum size allowed by WPA2, and change it either after each user connection, or periodically. This makes it difficult for an attacker to guess the key, assuming the usage of a secure Random Number Generator (RNG) algorithm in the system.
- *A revoked user*: Our proposed method does not actually address this issue, since the hardness of denying access of a revoked user relies on the difficulty of revoking a user in an ABE-based system (i.e., revoking the attributes associated to that user as proposed in [38]). Such an attribute revocation approach is neither practical nor scalable, since it requires an authority to periodically broadcast a key update information so that only the non-revoked users can update their keys and continue to decrypt messages.

## 10.2 Privacy and Security Considerations for the proposed MA-CP-ABE P-ABAC Scheme

In section 3.1.1 we have introduced a Multi-Authority Ciphertext-Policy Attribute-Based Encryption scheme which allows for Privacy-Preserving Attribute-Based Access Control.

During the setup phase, to preserve the security and trust of the system, issuing IdPs must never disclose their Master Secret Keys (MSKs). If the MSK of an issuing IdP is disclosed or leaked, the set of all attributes supported by the issuing IdP must not be used anymore by verifying IdPs.

During the issuance phase, the communication between the issuing IdP and the user must be performed through a secure channel (e.g. TLS transport mode) to avoid the eavesdropping of the credential issued to the user. Moreover, in order to support the expiration of credentials, the attributes should be defined by considering timestamps or version numbers, since MA-CP-ABE does not support policies with inequalities.

For what concerns the proving phase, the verifying IdP should include the issuance date in the access policy. Moreover, if using as a challenge a random value, this should not repeat in subsequent authentications. If using an ephemeral secret key, this should be generated randomly and also should not be repeated in subsequent authentications.

## 11 Conclusion

This deliverable entitled “Advanced Extensions: cryptographic attribute management, learning algorithms for complex ABAC reasoning and privacy awareness tool” follows and improves the architecture defined in D5.1 “Specification and Initial Design of the ABAC Infrastructure” and D5.2 “Full design and prototype of the ABAC infrastructure”. This document provides the description of the advanced cryptographic extensions that are employed in order to offer a state-of-the-art Privacy preserving attribute based access control solution. This includes recent advancements of the research community such as Attribute based encryption, cryptographic credentials stacks such as Idemix and U-Prove, and Trusted Execution environments. Furthermore, we describe the development of complex tools that allow users to: i) easily manage access control policies on the Service Provider side (Access Control Reasoning tool); ii) allow users and administrators to set their consent with regard to the reveal and transfer of identity attributes among entities of the ReCRED ecosystem; and iii) complex risk awareness tool that allow users to get indications with regard to the underlying risks when revealing and transferring identity attributes among the entities of the ReCRED ecosystem.

The results of this deliverable will be used as input for the integration activities and for the deployment of the pilots.

## 12 References

- [1] Android Open Source Project, "KeyGenerator", [Online]. Available: <http://developer.android.com/reference/javax/crypto/KeyGenerator.html> [Accessed 3-2016]
- [2] Android Open Source Project, "Cipher", [Online]. Available: <http://developer.android.com/reference/javax/crypto/Cipher.html> [Accessed 3-2016].
- [3] Dworkin, M. (2007, November). Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC.
- [4] Schaad, J., & Housley, R. (2002). Advanced Encryption Standard (AES) key wrap algorithm.
- [5] Housley, R., & Dworkin, M. (2009). Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm.
- [6] Android Open Source Project, "Android Keystore System", [Online]. Available: <https://source.android.com/security/keystore/index.html> [Accessed 3-2016].
- [7] Android Open Source Project, "KeyGenerator", [Online]. Available: <http://developer.android.com/reference/javax/crypto/KeyGenerator.html> [Accessed 3-2016].
- [8] Android Open Source Project, "Cipher", [Online]. Available: <http://developer.android.com/reference/javax/crypto/Cipher.html> [Accessed 3-2016].
- [9] "ICT facts and figures 2016," 2016 (2016) ict facts and figures 2016. <http://www.itu.int/en/ITU-D/Statistics/Pages/facts/default.aspx> .
- [10] "Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021 white paper," 2017 <http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html> .
- [11] A. Cassola, E.-O. Blass, and G. Noubir, "Authenticating privately over public Wi-Fi hotspots," in *Proc. of the 22nd acm sigsac conference on computer and communications security*, 2015, pp. 1346–1357.
- [12] WiFi Alliance, "WPA2 security now mandatory for Wi-Fi certified products," *Press Release*, 2006.
- [13] H. Boland and H. Mousavi, "Security issues of the IEEE 802.11b wireless LAN," in *Proc. of the canadian conference on electrical and computer engineering*, 2004, vol. 1, pp. 333–336.
- [14] A. Cassola, W. K. Robertson, E. Kirda, and G. Noubir, "A practical, targeted, and stealthy attack against WPA enterprise authentication." in *Proc. of the 20th annual network and distributed system security symposium*, 2013.
- [15] C. Pisa, A. Caponi, T. Dargahi, G. Bianchi, and N. Blefari-Melazzi, "WI-FAB: Attribute-based wlan access control, without pre-shared keys and backend infrastructures," in *Proc. of the 8th acm international workshop on hot topics in planet-scale mObile computing and online social neTworking*, 2016, pp. 31–36.
- [16] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. of the ieee symposium on security and privacy*, 2007, pp. 321–334.
- [17] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Advances in cryptology—eurocrypt*, Springer, 2011, pp. 568–588.
- [18] K. Frikken, M. Atallah, and J. Li, "Attribute-based access control with hidden policies and hidden credentials," *IEEE Transactions on Computers*, vol. 55, no. 10, pp. 1259–1270, 2006.

- [19] V. C. Hu, D. R. Kuhn, and D. F. Ferraiolo, “Attribute-based access control,” *Computer*, no. 2, pp. 85–88, 2015.
- [20] A. Sahai and B. Waters, “Fuzzy identity-based encryption,” in *Advances in cryptology–eurocrypt*, Springer, 2005, pp. 457–473.
- [21] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryption for fine-grained access control of encrypted data,” in *Proc. of the 13th acm conference on computer and communications security*, 2006, pp. 89–98.
- [22] R. Ostrovsky, A. Sahai, and B. Waters, “Attribute-based encryption with non-monotonic access structures,” in *Proc. of the 14th acm conference on computer and communications security*, 2007.
- [23] J. Lai, R. H. Deng, C. Guan, and J. Weng, “Attribute-based encryption with verifiable outsourced decryption,” *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, pp. 1343–1354, 2013.
- [24] M. Ambrosin *et al.*, “On the feasibility of attribute-based encryption on internet of things devices,” *IEEE Micro*, 2016.
- [25] J. Ning, Z. Cao, X. Dong, and L. Wei, “White-box traceable CP-ABE for cloud storage service: How to catch people leaking their access credentials effectively,” *IEEE Transactions on Dependable and Secure Computing*, 2016.
- [26] M. Jo, V. Odelu, A. K. Das, M. K. Khan, and K.-K. R. Choo, “Expressive CP-ABE scheme for mobile devices in IoT satisfying constant-size keys and ciphertexts,” *IEEE Access*, 2017.
- [27] M. Chase, “Multi-authority attribute based encryption,” in *Theory of cryptography conference*, 2007, pp. 515–534.
- [28] S. Muller, S. Katzenbeisser, and C. Eckert, “On multi-authority ciphertext-policy attribute-based encryption,” *Bulletin of the Korean Mathematical Society*, vol. 46, no. 4, pp. 803–819, 2009.
- [29] Y. Rouselakis and B. Waters, “Efficient statically-secure large-universe multi-authority attribute-based encryption,” in *Proc. of the international conference on financial cryptography and data security*, 2015, pp. 315–332.
- [30] P. Arana, “Benefits and vulnerabilities of Wi-Fi protected access 2 (WPA2),” *INFS 612*, pp. 1–6, 2006.
- [31] “IEEE standard for local and metropolitan area networks—port-based network access control,” *IEEE Std 802.1X-2010*, pp. 1–205, Feb. 2010.
- [32] J. Camenisch, M. Dubovitskaya, K. Haralambiev, and M. Kohlweiss, “Composable and modular anonymous credentials: Definitions and practical constructions,” in *Proc. of the international conference on the theory and application of cryptography and information security*, 2014, pp. 262–288.
- [33] J. Camenisch and E. Van Herreweghen, “Design and implementation of the idemix anonymous credential system,” in *Proc. of the 9th acm conference on computer and communications security*, 2002.
- [34] C. Paquin and G. Zaverucha, “U-prove cryptographic specification v1. 1,” *Technical Report*, Microsoft Corporation, 2011.
- [35] “OAuth: The ABC attack (the alice and bob collusion attack),” 2016  
<https://www.ietf.org/mail-archive/web/oauth/current/msg16767.html>.
- [36] “IEEE standard for information technology - telecommunications and information exchange between systems - local and metropolitan area networks - specific requirements - part 11: MAC and PHY specifications,” *IEEE Std 802.11-2007*, pp. 1–1076, June 2007.

- [37] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, “A digital fountain approach to reliable distribution of bulk data,” *ACM SIGCOMM Computer Communication Review*, vol. 28, no. 4, pp. 56–67, 1998.
- [38] A. Sahai, H. Seyalioglu, and B. Waters, “Dynamic credentials and ciphertext delegation for attribute-based encryption,” in *Advances in cryptology–crypto 2012*, Springer, 2012, pp. 199–217.
- [39] “LEDE project.” <https://lede-project.org/>.
- [40] “Charm: A framework for rapidly prototyping cryptosystems.” <https://github.com/JHUISI/charm>.
- [41] “Linux wireless - hostapd.” <http://linuxwireless.org/en/users/Documentation/hostapd/>.
- [42] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *Proceedings of the IEEE Symposium on Security and Privacy, SP’07*, pages 321–334. IEEE, 2007.
- [43] Chase, Melissa. “Multi-authority attribute based encryption.” *Theory of cryptography*. Springer Berlin Heidelberg, 2007. 515-534.
- [44] Lewko, Allison, and Brent Waters. “Decentralizing attribute-based encryption.” *Advances in Cryptology–EUROCRYPT 2011*. Springer Berlin Heidelberg, 2011. 568-588.
- [45] FIWARE Privacy Open RESTful API Specification [https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/Privacy\\_Open\\_RESTful API Specification](https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/Privacy_Open_RESTful_API_Specification)