Hindawi
Mobile Information Systems
Volume 2017, Article ID 6384186, 20 pages
https://doi.org/10.1155/2017/6384186

Research Article

Holistic Privacy-Preserving Identity Management System for the Internet of Things

Jorge Bernal Bernabe, Jose L. Hernandez-Ramos, and Antonio F. Skarmeta Gomez

Departamento de Ingenieria de la Informacion y las Comunicaciones, University of Murcia, Murcia, Spain

Correspondence should be addressed to Jorge Bernal Bernabe; jorgebernal@um.es

Received 7 April 2017; Revised 26 June 2017; Accepted 5 July 2017; Published 8 August 2017

Academic Editor: Michele Ruta

Security and privacy concerns are becoming an important barrier for large scale adoption and deployment of the Internet of Things. To address this issue, the identity management system defined herein provides a novel holistic and privacy-preserving solution aiming to cope with heterogeneous scenarios that requires both traditional online access control and authentication, along with claim-based approach for M2M (machine to machine) interactions required in IoT. It combines a cryptographic approach for claim-based authentication using the Idemix anonymous credential system, together with classic IdM mechanisms by relying on the FIWARE IdM (Keyrock). This symbiosis endows the IdM system with advanced features such as privacy-preserving, minimal disclosure, zero-knowledge proofs, unlikability, confidentiality, pseudonymity, strong authentication, user consent, and offline M2M transactions. The IdM system has been specially tailored for the Internet of Things bearing in mind the management of both users' and smart objects' identity. Moreover, the IdM system has been successfully implemented, deployed, and tested in the scope of SocIoTal European research project.

1. Introduction

Nowadays, a plethora of embedded and mobile devices can be accessed ubiquitously in different scenarios, such as transport systems, critical infrastructures, or smart cities. In order to deal with these applications, the *Internet of Things* (IoT) [1] is based on the notion of global connectivity to generate, process, and exchange large amounts of sensitive and critical data, which makes them appealing for attackers. In IoT, billions of interconnected "things" distributed across remote areas serve as a baseline for providing innovative services, which can be accessed not only through the Cloud, but also in a *Machine to Machine* (M2M) fashion [2]. M2M is considered as a key aspect for a broad adoption of the IoT, since M2M enables a direct communication among such *smart objects* [3] in an autonomous way. In such a distributed and dynamic environment, devices and services are exposed to additional threats that can compromise their data and, ultimately, the personal and private identity of the involved end users. Consequently, there is a strong need for not only adapting *identity management* (IdM) mechanisms to deal with user's identities, as it has been studied so far, but also allowing the management of smart objects' identities. In this sense, smart objects should be autonomous and independent entities with their own attributes and identity management mechanisms, which will allow them to preserve its owner's privacy during their operation.

Traditional privacy-preserving identity management solutions allow end users to manage their personal data for accessing certain services, by providing *user consent* mechanisms. Indeed, minimizing the disclosure of *Personally Identifiable Information* (PII) [4] is a basic requirement to realize the *Privacy by Design* (PbD) notions [5]. However, in IoT, a huge amount of smart objects are enabled to interact with each other, so an explicit user consent for each interaction is not feasible, due to scalability reasons. Furthermore, such smart objects could lack user interface, and consequently, human interaction should be maintained at the minimum. Additionally, while technologies such as the *Security Assertion Markup Language* (SAML) or OpenID [6] allow a selective disclosure of PII, these approaches are based on the presence of a *Trusted Third Party* (TTP) that needs to be queried during interaction between two entities or services, making the adoption of a real M2M approach difficult to be accomplished.

In order to address the challenges arising from the extension of identity management to any*thing* in our environment, this work proposes a holistic IdM system based on different emerging cryptographic technologies and approaches. In particular, the proposed IdM system follows a claims-based approach, which is built on top of the *Identity Mixer* (Idemix) technology [7] (from IBM) to provide additional means to deal with IoT scenarios where interacting entities can be smart objects, in addition to traditional computers. The proposed system endows users and smart objects with means to control and manage their private data, by defining *partial identities*, as a subset of identity attributes from their whole *virtual identity*. The use of partial identities aims to ensure a privacy-preserving solution with minimal disclosure of PII. Furthermore, unlike more traditional IdM approaches, the interaction between smart objects does not require an online TTP (typically an *Identity Provider* (IdP)), which is a valuable feature to foster the adoption of M2M approaches for the IoT. Moreover, the proposed solution relies on the Keyrock IdM system from the well-known FIWARE platform (https://www.fiware.org/). The main motivation to consider Keyrock is twofold: on the one hand, to support classic IdM operations and services, such as *Single Sign-On* (SSO) or *Identity Federation*, which are commonly used in Web or Cloud scenarios, where a claim-based approach is not required; on the other hand, Keyrock which is used as a repository of users and smart objects in which they are identified by using the *System for Cross-domain Identity Management* (SCIM) standard [8]. By this way, they are enabled to get Idemix credentials that are associated with SCIM identity attributes.

Furthermore, in order to demonstrate the potential of the proposed IdM system, this has been used as a mechanism for obtaining cryptographic credentials in a privacy-preserving way. In particular, a smart object can make use of its Idemix credential to derive *proofs*, in order to disclose only a subset of its identity attributes. In this way, users or smart object are enabled to use these proofs, in order to obtain security credentials for accessing IoT services. On the one hand, this has been integrated with our *Distributed Capability-Based Access Control* (DCapBAC) approach [9], as a lightweight and distributed authorization model to be used in IoT environments. In this case, the identity attributes that are disclosed by using a specific proof are employed during the authorization process based on the *eXtensible Access Control Markup Language* (XACML) [10] to obtain the DCapBAC token. On the other hand, this has been used to obtain cryptographic keys based on the *Ciphertext-Policy Attribute-Based Encryption* scheme [11], as a flexible approach for scenarios where information needs to be shared with groups of users or smart objects. Therefore, the CP-ABE keys are associated with the attributes that are disclosed by using a specific proof. In addition, the resulting IdM system is framed within a security and privacy architecture [12] based on the *Architecture Reference Model* (ARM) [13] from the IoT-A project. In particular, this represents the instantiation of the IdM functional component, as well as its interactions with other components. Furthermore, in order to demonstrate its applicability in different scenarios, the proposed solution has been designed, developed, and deployed in the scope of the European project SocIoTal (http://sociotal.eu/). To the best of our knowledge, the proposed IdM system is the first approach that aims to provide a holistic privacy-preserving identity management system for IoT, by providing a unified solution to heterogeneous scenarios in which privacy must be ensured.

The rest of the paper is structured as follows. Section 2 aims to provide a description of IdM related works for the IoT and the most predominant IdM systems that are used currently. Section 3 presents some of the main challenges related to the extension of identity for smart objects, as well as an overview of the proposed approach. Section 4 describes the SocIoTal security and privacy framework in which the IdM system's functionality is framed, and a detailed description of the proposed system's actors and interactions is given in Section 5. Section 6 provides a set of experimental results by considering different functionality of the IdM system, and Section 7 compares qualitatively such system with other more established IdM approaches. Finally, Section 8 concludes the papers and provides an outlook of our future work in this area.

2. Background and Related Work

2.1. Related Work. The realization of a real Internet of Things requires the adoption of identity management models that can be adapted to identify any smart object connected to the Internet. Furthermore, such model must take into account IoT's inherent security and privacy requirements, providing a high degree of scalability (to deal with crowded scenarios) and flexibility (by using advanced schemes beyond the use of simple identifiers or IP addresses). These issues have attracted a growing interest from academia [14–16] during last years. In this sense, while there are already widely deployed identity management approaches in Web or Cloud environments, their applicability in IoT scenarios has not been demonstrated. Indeed, traditional identity management systems, such as SAML [17] or OpenID [6] based approaches, require a typical IdP (acting as a TTP) entity, which is responsible for managing and generating on-demand access tokens to enable a secure and trusted communication between interacting entities. However, this TTP hampers the adoption of security and privacy-preserving mechanisms for M2M scenarios that are required in IoT, since the IdP is required to be online to complete the transaction between the parties.

Moreover, traditional IdM systems do not provide means to their users to deal with the data *minimization principle*, which is a core aspect of the recent *General Data Protection Regulation* (GDPR) (Article 5. Principles relating to processing of personal data) (http://ec.europa.eu/justice/data-protection/reform/files/regulation_oj_en.pdf). This basic privacy principle is directly related to the partial identity notion. Some EU projects, such as SWIFT [18], already delved into this concept. In particular, they proposed an instantiation by defining an infrastructure based on the use of SAML and XACML technologies. Other approaches [19] use X.509

certificates as credentials to model the real identities and SAML security tokens to encode partial identities to be used afterward to prove the possession of certain attributes. Thus, users have control over the attributes which are disclosed to each service. The usage of a certificate avoids the involvement of the IdP in each communication, and therefore, different transactions cannot be linked by this entity. However, the usage of certificates does not preserve anonymity since entities are unequivocally identified in the certificate that is entirely disclosed to the other party.

In contrast to these approaches, *Anonymous Credential Systems* (ACS) [20], such as Idemix [7] or U-Prove [21], allow users to present cryptographic proofs, instead of the whole credential, proving the possession of certain attributes or claims. These systems enable a selective disclosure of identity attributes to achieve a privacy-preserving identity management approach. Indeed, a user or entity can prove a specific set of properties associated with a subset of identity attributes, without disclosing the content of such attributes itself. Even though some important initiatives, such as PrimeLife (http://primelife.ercim.eu/) or ABC4Trust [22] have analyzed the use of ACS-based and privacy-preserving identity management systems, the IoT paradigm demands adapted solutions to ensure a seamless development of an IP-based approach [23]. Unlike in current Internet scenarios, common IoT use cases are based on a huge amount of heterogeneous smart objects interacting, and consequently, communications protocols and implementation need to be adapted. Current solutions must evolve to enable an IdM approach for IoT without the requirement of an online TTP, while security and privacy (including data minimization and unlikability principles) are met.

In this direction, [24] proposes a set of trust-enhancing security functional components for the IoT resolution infrastructure, in order to provide basic security and privacy aspects in IoT communications. Furthermore, [25] describes a PKI authentication scheme and discusses different practical solutions for realizing users' anonymity, delving into the complexity related to the design of privacy-preserving solutions based on such schemes. Moreover, [26] presents a distributed target-driven anonymous authentication protocol for IoT applications. This proposal is based on a multishow credentials system, which is used by users to authenticate anonymously. In addition, in [27] provides an authorization framework based on SAML assertions that are obtained from decisions made by an engine based on XACML. These assertions are used to get access to an IoT device through the use of symmetric key cryptography. However, such a work lacks evaluation details.

A common aspect of these proposals is that they are based on the use of traditional symmetric and public key cryptography to identify users and smart objects. In contrast, our approach is based on the concept of virtual/partial identity to identify any entity that is intended to participate in the IoT ecosystem. In this way, users and smart objects are enabled to use a subset of their whole identity for accessing to IoT services. The instantiation of such concept has been materialized through the use of Idemix as the most ACS representative example and deployed on the European platform FIWARE, as will be described in Section 5.

2.2. Background on Traditional IdM Systems. *OpenID* [6] is an open standard that enables users with a single identity for being used across disparate Internet services, without the need to hold and remember multiple identifiers and login information. Users can be authenticated by certain cooperating sites (i.e., *Relying Parties*) using a third-party service, reducing the need of *Service Providers* (SPs) to employ their own systems. It uses HTTP cookies to support SSO across different SPs. The OpenID identifier is in the form of a URI that is issued by OpenID providers (IdP). Major companies act as OpenID providers for their customers and provide them with an OpenID identifier, but OpenID accounts are not limited to commercial IdPs; independent IdPs are also available. OpenID requires end users to provide the same OpenID identifier in every SP, so they can be tracked. It means that users have to create and manage different OpenID accounts to preserve their privacy.

SAML [17] is an OASIS standard for conveying security information across entities. It is foremost employed to transfer user authentication or authorization information within assertions from one communication partner to another. It also defines four XML-based mechanisms: assertions, protocols, bindings, and profiles. SAML is currently mostly used for SSO and attribute-based authorization. In these cases, authentication assertions are used to transfer authentication information from the authenticating entity (e.g., an IdP) to the requesting entity (usually a SP) to avoid a new authentication of the user at the SP. For attribute-based authorization, attribute assertions are used to send identity attributes information from an attribute authority, so authorization decisions can be made by using such attributes. However, in both cases, the SP redirects the user to be authenticated by the IdP, and the user has no control over which PII from the IdP is disclosed to the SP.

OAuth [28] is an authorization protocol that allows third parties applications or clients to access resources owned by a user (hosted in trusted applications) without the need to give or know the user's credentials. That is, third-party applications can access resources owned by the user, but these applications do not know the authentication credentials. In this way, through the OAuth 2.0 authorization framework, an application can obtain limited access to a resource, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and the HTTP service or by allowing the third-party application to obtain access on its own behalf. OAuth 2.0 introduces an authorization layer and separates the role of the client from that of the resource owner. In this way, the client requests access to resources which are managed by the resource owner and hosted by the resource server and issued a different set of credentials compared to those of the resource owner.

Nowadays, these technologies are used as the basis to build common and widely deployed IdM systems, such as Shibboleth or Keyrock.

Shibboleth [29] is an open source project that provides a federated identity solution aimed at allowing sites to make authorization decisions to protected online resources in a privacy-preserving way. As other approaches, Shibboleth defines the IdP and SP actors, so users are redirected to the IdP of the organization where the user belongs to, in order to get an authentication assertion, which is in turn used to gain access to a resource in the SP. It provides a HTTP-based SSO approach, in which each organization can use a different authentication mechanism. Shibboleth is a well-known IdM system that is commonly used by academic organizations to deal with identity federation aspects. However, currently it is not under consideration as a candidate IdM approach to be deployed in IoT scenarios.

Keyrock IdM (https://catalogue.fiware.org/enablers/identity-management-keyrock) is an open source implementation of the IdM system defined in FIWARE, which is a European middleware platform that promotes the development and global deployment of applications for the Future Internet. FIWARE delivers reference architecture, as well as the specification and implementation of different open interfaces called *Generic Enablers* (GEs). Among the FIWARE GEs, the identity management GE relies on standard protocols, such as SAML and OAuth, to provide authentication and authorization features, which allows managing users' access to networks, services, and applications. The IdM GE is also responsible for the user profile management, as well as SSO and identity federation across different service domains. Keyrock relies on the OpenStack IdM implementation called Keystone (https://docs.openstack.org/developer/keystone/), extending Keystone by providing an implementation of the SCIM standard. SCIM is intended to reduce the cost and complexity of user management operations through a common user schema, extension model, and REST API with a rich, but simple set of operations.

These features from Keyrock have been leveraged by our proposed IdM system. In particular, the Keyrock implementation has been used as a repository of users and smart objects, so security credentials can be associated with the SCIM identity attributes defined in it. In fact, Keyrock has been used as a standalone IdM system and at the same time as a complementary approach for the definition of our privacy-preserving Idemix-based IdM system. This integrative solution is intended to provide an open and holistic IdM system for the IoT by using a reference platform at European level, which fosters its application and adoption in different IoT environments.

2.3. Background on Anonymous Credential Systems. Anonymous Credentials Systems (ACS) [20] allow an IdP to issue a credential to a user, to be used afterwards, to authenticate against a SP. The credential may contain user's identity attributes (e.g., address and age) or be related to a smart object (e.g., hardware properties or manufacturer). Within the credential, the user or smart object can later prove to a third party that he possesses a credential containing a given attribute without revealing any other information stored in the credential. This proving process is usually carried out through the use of a cryptographic proof, which can be derived from the credential.

U-Prove [21] is an anonymous credential system developed by Microsoft for claims-based identity management. It provides strong security, scalability, and privacy and allows for claims to be efficiently tied to the use of smart cards. A U-Prove token is a digitally signed container of attribute information of any type. Each token is issued to a *Prover* generated by an *Issuer* during the *issuance* protocol. When using a U-Prove token, the *Prover* applies the token's private key to a message to create a presentation proof for the *Verifier*. Such proof represents a proof of possession of the private key as well as a digital signature of the Prover on the message.

Idemix [7] is an ACS developed by IBM that allows users to minimize the personal data they have to reveal in electronic communications. Idemix defines the roles of Issuer, Recipient, Prover, and Verifier. Credential *issuance* is carried out between an *Issuer* and a *Recipient* (user), following a protocol which ends up with the latter having a credential. This credential consists of a set of attribute values, as well as cryptographic information that allows the credential's owner to create a proof of possession. In this way, when creating a proof, the user (acting as a *Prover*) can prove the possession of a certain credential to a SP (acting as a *Verifier*). Several zero-knowledge proofs are performed to convince the SP about the possession of such credential by making use of the CL signature scheme [30]. Furthermore, Idemix technology also allows proving the possession of several credentials at the same time and stating different relationships among the attributes contained in such credentials.

Although the main purpose of U-Prove is similar to Idemix, and selective disclosure can be done similarly, there are different cryptographic differences between both approaches. Unlike U-Prove, users in Idemix need only one secret key (known as the *master key*) corresponding to all credentials. Whereas U-Prove credentials are revealed every time they are used, an Idemix credential is never disclosed to a verifier. This means that unlinkability in U-Prove can be achieved only using different credentials, whereas Idemix supports multishow unlikability against the SP. Regarding performance comparison, U-Prove is more efficient than Idemix when it comes to selective disclose of attributes. The multishow unlinkability property of the Idemix technology leads to a reduction of his performance when it is compared to U-Prove. The issuer signature is partly randomized and the remaining part is proved by using a zero-knowledge proof to achieve unlinkability. This process is not carried out in U-Prove that only requires computation to hide the undisclosed attributes.

Idemix technology has been widely used in different European projects, such as ABC4Trust or PrimeLife. Due to its privacy properties and extensive acceptance, our proposed IdM system is based on Idemix to enable privacy-preserving access to IoT services. Specifically, as an instantiation of these services, Idemix has been employed to obtain on-demand access credentials, specifically by using DCapBAC and CP-ABE approaches. The design, implementation, and deployment have been carried out under the umbrella of the SocIoTal project and will be explained in Section 5.

3. IoT Identities

Unlike current Web or Cloud environments, identity management must cope with unique challenges in the IoT paradigm. On the one hand, any smart object that is connected to the Internet must be identifiable to be able for interacting with other entities. This identification scheme must go beyond the use of simple networking identifiers or IP address, but also through the use of specific features or attributes that identify the object, such as its manufacturer, owner, or hardware properties. Moreover, smart objects should be able to identify themselves through temporal identities to be used according to the context, such as its location. On the other hand, smart objects could have to act on behalf of their owner. In this sense, delegation schemes are important to authorize devices to perform actions on behalf of the user making use of a certain partial identity according to the context.

The identity management approach should be *distributed* in order to authenticate smart objects among each other but, at the same time, *centralized* enough to be able to establish a hierarchical approach where identities credentials can be issued and authenticated securely enabling a global digital trust environment. At the same time, users and objects can constitute groups or communities with some interest in common. In these scenarios, they could also act with different partial identities according to the context, as well as to use different enforced authorization rules for each of the group they belong to. These issues are exacerbated when privacy concerns are also to be addressed, in order to enable a privacy-preserving but accountable IdM system for the IoT.

3.1. Object Naming, Addressing, and Discovery. An essential feature for the IoT is the inclusion of an infrastructure to make smart objects addressable, named, and finally discovered. Unlike in small-scale application silos making up an *Intra*net of Things [31], applications and services cannot be configured with respect to a fixed set of services. This is mainly due to the inherent dynamism of the IoT ecosystem resulting from the mobility of smart object, as well as the changing availability of services due to constraints of the underlying resources and devices. Therefore, a real need exists for a suitable infrastructure to be in place that allows addressing, naming, and discovery of IoT services:

(i) IoT addressing: an IoT address refers to an identifier of a smart object and/or its virtual representation. This feature entails the assignment and management of addresses/identifiers for smart objects.

(ii) IoT naming: it refers to mechanisms and techniques for assigning names to objects and supporting their resolution/mapping to IoT addresses. IoT naming provides the means to identify smart objects through a name resolution mechanism according to a naming system. Additionally, names can be organized according to taxonomies or classifications in a hierarchical fashion and, consequently, to be used for groups of smart objects.

(iii) IoT discovery: it refers to the process of locating and retrieving IoT resources in the scope of a large and complex space of smart objects.

The three previous concepts are closely related, given that the adherence to certain choices and solutions (e.g., standards, mechanisms, algorithms, and tools) for one area (e.g., choice of addresses/identifiers) can directly affect the respective choices and solutions in the other areas (e.g., naming system used). As a consequence, the consideration of solutions for one area cannot be seen as isolated from the others.

Although several proposals have been introduced for all these areas, there are still significant challenges to be addressed. On the one hand, regarding identification, a globally unique identification solution, is needed, supporting the requirement for unique namespaces and the description of the wide diversity of smart objects.

On the other hand, for discovery issues, it is necessary to address some requirements regarding the visibility of smart objects' properties across different granularities, as well as the addition of semantic and contextual information for the discovery process. This information would enable to cope with mobility issues during the process by providing a more expressive and rich discovery process of IoT resources.

In this context, the Handle System [32] is getting attraction for IoT naming, resolution, and discovery. The unique identifier (UID) given to a smart object can be represented by a Handle ID (HID). In Handle, the identifier consists of a prefix and a suffix as depicted in Figure 1. The prefix is a unique integer managed globally by the Handle Global Registry, whereas the suffix represents a string associated with the local domain where the smart object is deployed. The suffix can be based on an URI, URN, Electronic Products Code (EPC), Digital Object Identifier (DOI), MAC address, IMEI, HashValue, and so on. Indeed, the suffix can be a combination of them, for instance, a URI that contains a DOI or a MAC address as part of the URI. Having HIDs as identifiers allows IPv6 address resolution. Indeed, Handle provides a mechanism for linking multiple network addresses with HID for the same smart object. Thus, during the bootstrapping or setup phase, each smart object can be assigned a HID algorithmically. For instance, in a smart building the local prefix of the HID could be assigned according to hierarchical URIs for each floor, room, and smart object. Our IdM allows representing the UID (for users and smart objects) as any string, so that the HID is a suitable option that can be used as identifier in our IdM system.

3.2. Privacy-Preserving Attribute-Based Credentials for IoT. Traditional identity management systems usually assign a unique number to identify the identities. Although it makes easier the management, this unique number also leads to a lack of privacy for user that can be traced easily. The service provider (SP) could save all the tokens and user credentials that are presented to it and then link them altogether. The user's public keys as well as the issuer signature are left behind at each service provider that is accessed, leaving a digital trail that can be then used to link its behavior and make up user's profiles such as dossiers for each individual with his or movements, habits, or preferences.

As already mentioned, ACS try to solve these issues. A credential can be defined as a container of attributes that is

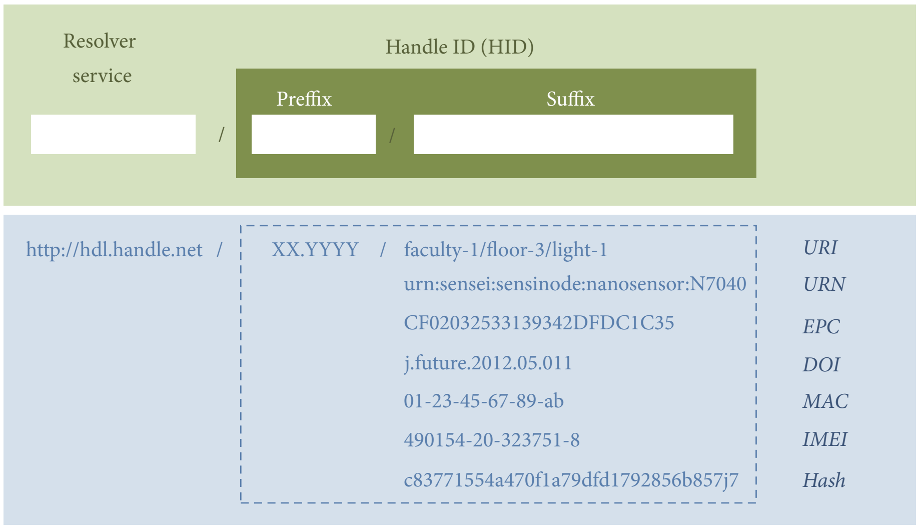


Figure 1: Unique identifier (UID) for IoT based on Handle.

employed as a certificate to convince others about the facts attested by it. This credential is issued by a TTP (i.e., an issuer) to the Recipient entity (either a user or smart object). When using an ACS, the Recipient can employ their credentials to derive partial identities as proofs of possessing a particular subset of the encoded attributes in the credential. These partial identities are, in fact, cryptographic proofs that can be presented to a SP or directly to another smart object (acting as a verifier) maintaining the credential itself undisclosed. Therefore, data minimization is provided for the Recipient to access a service, while the SP cannot impersonate it, since the credential is not disclosed. A credential is defined by means of a specification structure that specifies the list of attribute types to be encoded in the credential. In IoT, this structure should be defined preferable in JSON rather than in XML. There is a separation of the credential structure that is the public part and the credential data which is private to the Recipient.

Figure 2 shows a graphical representation of these concepts. Our IdM system manages the user's attributes as it is done in any traditional IdM. In addition, the system can manage directly smart objects' identities and their attributes. At the same time, the IdM system acts as a *Credential Issuer*, in charge of generating the Idemix credentials for both the users and their associated smart objects, according to the attributes hold in the IdM. Each user has associated one or more attributes among those defined in the SCIM standard (e.g., ID, name, address, domain, and email), whereas a smart object has its particular set of attributes, such as ID, vendor, manufacturer, date, and model. The association between the smart object and its owner is done by means of the attribute *owner* maintained in the smart object. It should be noted that the ID attribute is represented by the Handle ID (HID), which was previously described. In this way, when a user or smart object tries to get access to a particular IoT service, he makes use of its partial identity to demonstrate the required attributes, as will be explained in Section 5.

4. Security and Privacy Framework Overview

The proposed IdM system has been designed under an IoT security and privacy framework based on the *Architectural Reference Model* (ARM) [13], which was proposed by the IoT-A project. In this sense, such framework represents a simplified view of our security and privacy architecture for the lifecycle of smart objects (ARMY) [12] that is intended to provide a comprehensive view of security and privacy implications in IoT. ARMY represents an instantiation of the Functional View of the *Reference Architecture* (RA), and an extension through the addition of two new functional components: the Context Manager and the Group Manager. Both elements have been included in order to complement the functionality that is already provided by other components within the security functional group. The framework has been designed, developed, and deployed in the scope of the SocIoTal project. Figure 3 shows the main components of the security framework, which are explained below.

The *Authentication* component is intended to ensure only legitimate users and smart objects are able to access a specific IoT service. In the context of the SocIoTal project, this component's functionality has been instantiated through the use of certificate-based public key cryptography to be used in scenarios and use cases in which privacy is not addressed. In particular, it has been based on the use of *Elliptic Curve Cryptography* (ECC) and integrated with *Transport Layer Security* (TLS) [33], as well as DTLS [34] in case of communications based on the *Constrained Application Protocol* (CoAP) [35].

The *Authorization* component is responsible for making and enforcing authorization decisions by integrating different access control technologies and approaches. On the one hand, it follows a policy-based scheme to define access control policies that are evaluated to come up with a decision. On the other hand, it follows a token-based approach generating authorization tokens according to the decision. In particular,

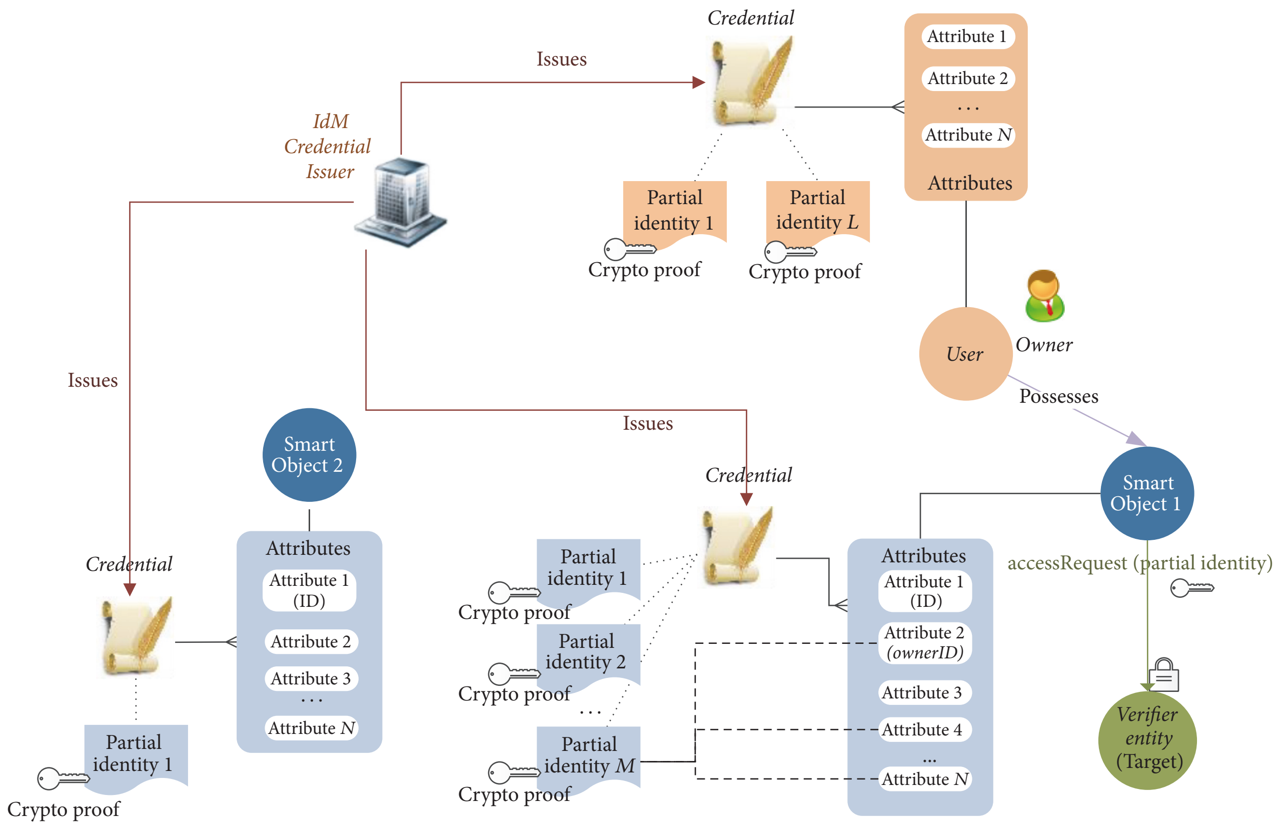


FIGURE 2: IoT partial identities representation.

such functionality has been instantiated through the use of the XACML standard to define authorization policies and the DCapBAC approach [9], which is proposed as a flexible and lightweight to be used in different heterogeneous IoT scenarios.

The *Group Manager* component aims to address security and privacy concerns in common crowded IoT scenarios, where data needs to be shared or outsourced to groups of users and smart objects. This component's functionality has been realized through the use of the CP-ABE scheme [11] together with signatures schemes to guarantee data integrity. In this way, data is encrypted under a policy of attributes, while secret keys are associated with sets of identity attributes. Only those users or smart objects that satisfy the encryption policy will be able to access such data. Consequently, a data producer can exert full control over how the information is disseminated to other entities, while a consumer's identity can be intuitively reflected by a certain secret key.

The *Key Exchange and Management* (KEM) component is focused on assisting interacting entities with key management tasks (e.g., to establish a security context) to enable secure communications among them. Additional functionality includes generating and delivering cryptographic keys, including symmetric and public keys, as well as CP-ABE keys that are used by the Group Manager component

The *Context Manager* component is one of the key components in the framework. It aims to realize an adaptive security and privacy approach for the IoT. In particular, it is intended to maintain contextual data so other components can adapt their behavior accordingly. For example, a user or smart object can be identified with a different partial identity according to context, or authorization decisions can change dynamically depending on the context detected by a smart object.

The *Trust and Reputation* component aims to define suitable trust and reputation models that are able to be integrated in the IoT ecosystem, in order to complement the cryptographic approach that is used by other security components. The set of trust scores obtained from such models are then used by other security components to provide an additional fine-grained level of security and privacy. Trust scores are also used to manage and share data within groups of users or smart objects. This component's functionality has been specifically instantiated by using a multidimensional trust model based on fuzzy logic that was also integrated with the DCapBAC approach for authorization purposes [36].

The *Identity Management* (IdM) component is responsible for managing the identity of users and smart objects. In particular, it aims to complement the functionality that is provided through the Authentication component, by enabling a scalable and privacy-preserving identity management mechanism based on the partial identity concept. Its functionality has been realized by using the Idemix technology to endow users and smart objects with mechanisms to ensure anonymity, which is done mainly issuing pseudonyms and proof of credentials for data minimization.

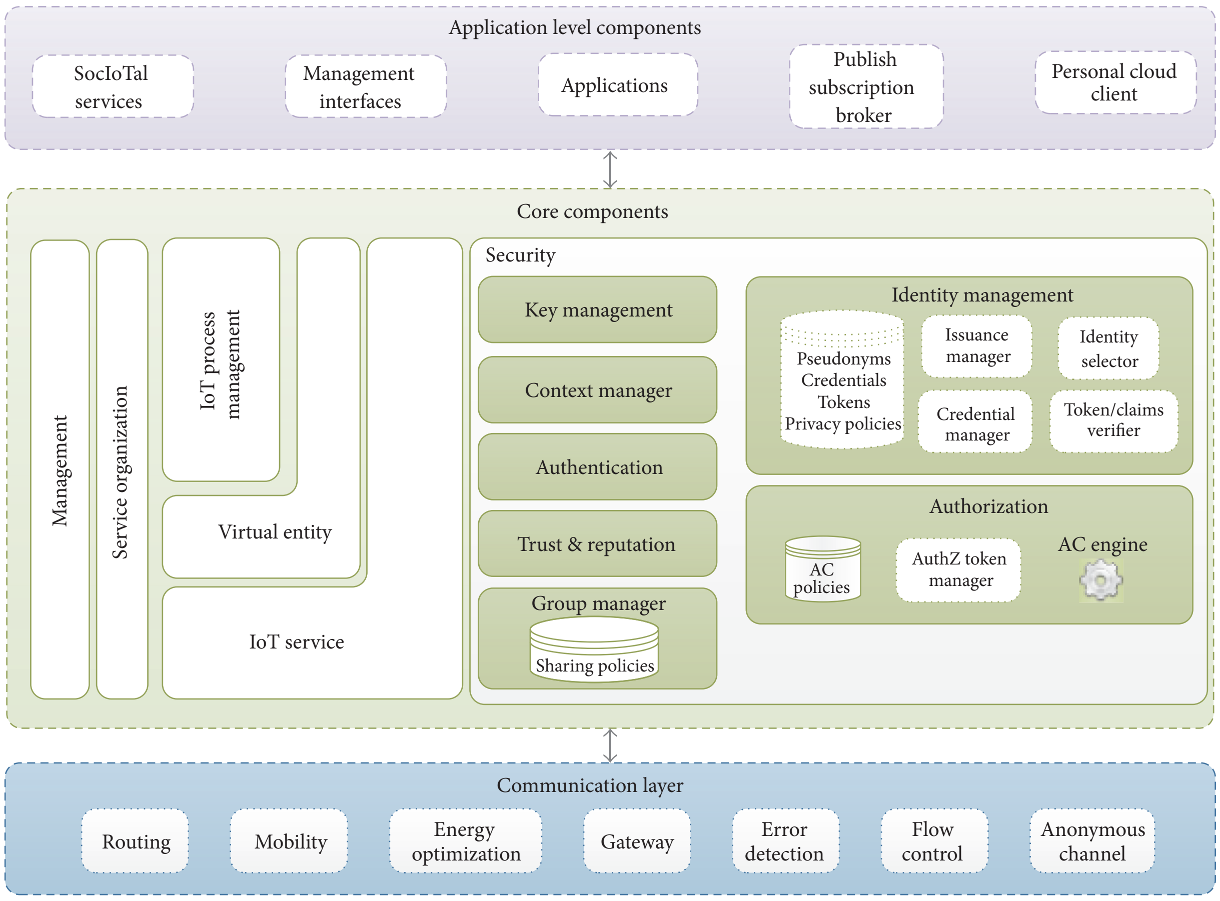


Figure 3: Security IoT framework based on ARM.

The IdM component is the cornerstone of the SocIoTal security and privacy framework and the core element of the proposed IdM system. It represents the integration of Idemix technology with a reference European IoT platform, FIWARE, to manage users and smart objects identity to access IoT services in a privacy-preserving way. Furthermore, it has been integrated with DCapBAC and CP-ABE approach, so entities are enabled to get authorization tokens and group keys while their privacy is still preserved. A more detailed description of the proposed IdM system is provided in the next section.

5. Holistic Identity Management System

In traditional IdM scenarios, organizations offer services that are managed and protected by SPs. In such scenarios, users' authentication is done by means of IdPs that assert identities to end users. Then, users present such assertions (e.g., based on SAML or OAuth) to obtain access to a resource at the SP, meaning that a trust relationships between SP and IdP is required. Unlike this classical web approach, in the IoT, the user role can be also played by a smart object, and the SP can be represented directly by another IoT device holding the resource being accessed.

The following subsections define the proposed holistic IdM system, including a detailed description of required deployment entities and interactions to accomplish such functionality.

5.1. IdM Actors. Figure 4 shows the main deployment entities involved in the different processes and interactions that are defined in our IdM system. These deployment entities instantiate and implement the IdM functional component of the SocIoTal security and privacy framework. As shown, in addition to more typical user management operations, the proposed IdM system aims to provide functionality related to authentication, authorization, and attribute management, as well as credential and cryptographic keys provisioning. The following subsections define each of these deployment elements including their features and interactions.

5.1.1. Subject. The *subject* represents a user or smart object that requires to access an IoT service in a privacy-preserving way, by ensuring the data minimization principle. The subject can get Idemix credentials from different *issuers*, so it is able to decide which particular information from which credential should be presented to a certain *target* service (acting as a *verifier*). In traditional web scenarios, the subject represents a user, but in IoT it represents a broader concept that refers to any kind of smart object. A subject plays the Idemix *Recipient* role, since it asks for credentials from an issuer. When the

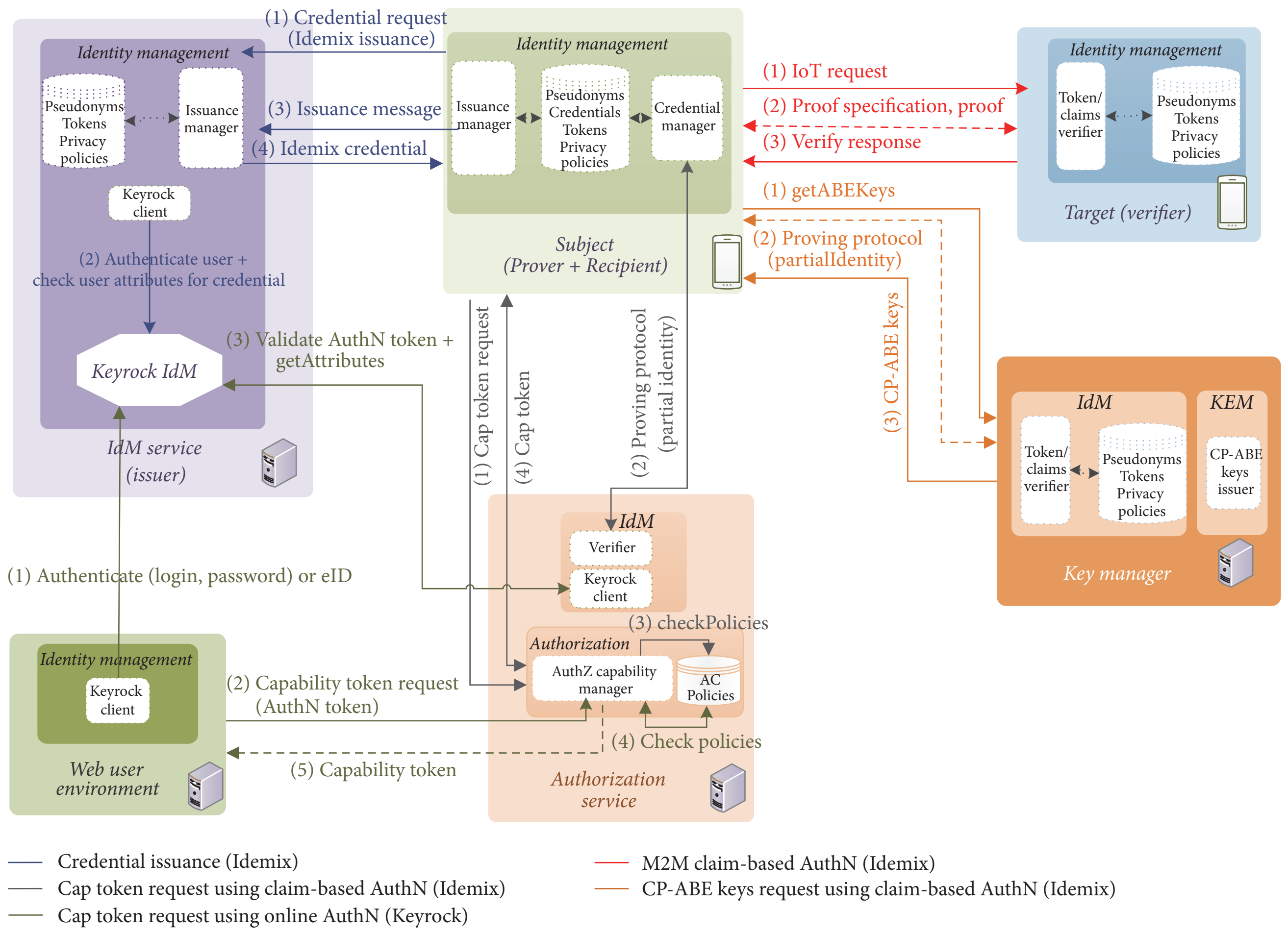


FIGURE 4: IdM main interactions.

subject obtains the credential, it can derive cryptographic proofs to prove certain attributes (or attributes statements) against an IoT service, playing the role of *prover*. In this case, the IoT service acts as verifier of the attributes and statements that are included in such a proof.

5.1.2. IdM Service. As already mentioned, our IdM system integrates FIWARE Keyrock IdM (supporting classic IdM features, such as SSO) and provides additional functionality by including new privacy-preserving features, which are built on top of Idemix technology. Towards this end, unlike in Keyrock, our IdM system supports additional attributes to manage smart objects' identity that are not covered in the SCIM model. While the IdM Service plays the traditional role of IdP and Attribute provider, it also acts as an Idemix *Issuer*. The issuer is in charge of issuing Idemix credentials to subjects, after checking the correctness and veracity of the information provided during a previous authentication process. In this sense, the IdM support different ways of authentication.

The Idemix credential structure is composed of the following attributes, which are taken from the SCIM standard. In Idemix, an attribute is defined by a tuple that consists of a name, value, and type $a = (n, v, t)$, where *name* can be Id, userName, domainId, name, active, email, nickName, country, locality, postalCode, streetAddress, department, organization, and phone and *type* can be integer, string, date, and enumerations, among others.

It should be noted that current Keyrock implementation does not support the management of all these attributes, so our IdM system extends Keyrock implementation. In addition, since SCIM does not include attributes to manage smart objects, our approach extends the SCIM data model to deal with smart object attributes, such as owner (i.e., the ID of the device's owner), vendor, manufacturer, manufacturing date, and model.

5.1.3. Authorization Service. In our IdM system, authorization decisions are not directly made by the SP or target entity that is accessed. Instead, this task is delegated to a separate *Authorization Service* that can manage the access control to different entities. The Authorization Service is in charge of generating DCapBAC tokens, that is, authorization assertions containing the access rights that a subject entity has over a resource, which is hosted by a target entity. These tokens are obtained by the subject to access directly to the target, which only needs to validate the token, since the decision was previously made when it was generated.

The Authorization Service provides a *Policy Decision Point* (PDP) to evaluate access requests against authorization policies that are previously defined in the Web User Environment. It is based on the XACML standard, so access control policies are handled and evaluated upon an access request by the PDP to make the corresponding authorization decision. Such decisions are based on the identity attributes of the requesting entity. These attributes can be proved directly by this entity during the token request through the use of Idemix, or they could be obtained from the IdM Service in case of Keyrock authentication.

5.1.4. Target. The *target* represents the IoT service that is accessed by a subject, and it acts as an Idemix *Verifier*. It protects access to an IoT service data by requiring the subject to satisfy certain identity attributes in their credentials. To this aim, the verifier presents a *policy* to the subject indicating which data from the credential needs to be disclosed, in order to obtain access to the IoT service. Indeed, it includes which credentials and attributes from which issuers are required or even which conditions have to be fulfilled by the attributes. Based on this policy, the subject generates a *proof* from its Idemix credential containing the required attributes as well as the cryptographic evidence to be sent to the verifier.

In IoT, unlike in the current Internet, the required process to discover which particular attributes are needed could require autonomous IoT operation, without user intervention. Devices should be given flexible and scalable mechanisms to discover these attributes, and the CoRe Link Format [37] can be used for this purpose. In this way, the subject can perform a multicast CoAP request to/*.well-known/core* and the Resource Directory can answer with information about the devices resources. The information format is compliant with the Link Format [38], but extending it in order to describe the attributes that are going to be needed to access to such a resource.

5.1.5. Web User Environment. Keyrock is split into two components, the web-based front-end (Horizon) and the RESTful back-end (Keystone), which is the open source OpenStack IdM. Our IdM system does not relies on Horizon, as SocIoTal project already defines its own *Web User Environment*. This component is intended to provide graphical interfaces to manage user attributes. In addition, the Web User Environment acts as *Policy Administration Point* (PAP), allowing users to define and manage XACML authorization policies, which are used by the Authorization Service to make authorization decisions.

5.1.6. Key Manager Service. The *Key Manager Service* is responsible for generating cryptographic keys associated with legitimate and authenticated users or smart objects. In particular, this has been used as a service for the generation and delivery of CP-ABE keys, in order to obtain confidentiality when the information needs to be outsourced to groups of entities. In CP-ABE, data is encrypted under a combination or policy of identity attributes, while a CP-ABE key is associated with a set of attributes. Therefore, similar to the Authorization Service, when a subject tries to obtain a CP-ABE key, the Key Manager Service can obtain the subject's attributes from the IdM Service or directly from the cryptographic proof that is contained in the request, in case of using Idemix for a privacy-preserving authentication. In this way, users and smart objects can obtain a CP-ABE key associated with a partial identity, which is demonstrated by Idemix technology.

5.1.7. Revocation Authority. Finally, although it is not shown in Figure 4 for the sake of clarity, the IdM system is also equipped with a revocation service that is responsible for revoking issued credential, preventing their further usage to generate a presentation token in case the attributes are no longer valid. This revocation entity is also needed to deal with credential termination at the end of the identity lifecycle. In conventional credential systems, revocation status of a credential is verified just by simple lookup of a revealed credential specific identifier in a list, for example, Certificate Revocation List (CRL) [39]. In Idemix the revocation is done based on accumulators which uses the membership proof for white-listing. The revocation authority accumulates all valid revocations into a single value and publishes that value. Then, users can prove that their credential is still valid, with a zero-knowledge proof that demonstrates that the credential's revocation is contained in the published whitelist accumulator. For further information about revocation, the reader is referred to Idemix library specification [40].

5.2. IdM Interactions. After describing the main actors of our IdM system, this subsection aims to provide a detailed explanation of the required interactions among such entities, in order to realize the proposed functionality. In this way, Figure 4 provides an overview of the main processes (by using different colors), which are explained below.

5.2.1. Basic Authentication and Authorization Process. As already mentioned, our IdM system supports different authentication mechanisms, including the simplest (but still used) password-based approach together with the use of authentication bearer tokens. In this case, any subject in possession of a bearer token can employ it to obtain access for the IdM Service or a target resource, without the need to prove the possession of a cryptographic key. In order to avoid the abuse or misuse of tokens, the tokens can be sent securely using the DTLS protocol. To perform this basic authentication mechanism, the IdM Service contacts the Keyrock IdM, which in turn relies on Keytone.

Furthermore, the IdM system allows a subject to authenticate against the IdM Service by using certificate-based public key cryptography. This feature is provided by the IdM Service that verifies the certificate against the one that is already stored for the entity registered in Keyrock. In case the subject is not registered previously in the IdM Service, it can be done through either the API or the Web User Environment.

Once authenticated, the IdM Service provides the subject with a Keystone authentication bearer token that can be used to perform actions against the IdM Service (such as user management) or access to other target services by attaching the bearer token as a header in the HTTP request. In such a case, the target must contact the IdM Service to verify that the token provided by the subject is valid

5.2.2. Credential Issuance Process. In addition to the traditional password-based authentication mechanism, our IdM system enables a privacy-preserving authentication approach, which is built on top of Idemix technology. Idemix is based on two main protocols: *credential issuance* and *proving*. The former is used by the subject to obtain a credential, while the latter is employed to prove the possession of a certain credential when accessing a target's resource in a privacy-preserving way. According to the required interactions that are shown in Figure 4, below we provide a description of them:

(1) The subject requests a credential to the IdM Service, which acts as an Idemix Issuer. For this purpose, it presents its certificated to identify again the issuer.

(2) The IdM Service contacts the Keyrock IdM in order to validate the attributes for which the subject applies for a credential. It should be noted that Keyrock is responsible for validating the authenticity of subject's attributes prior the credential issuance.

(3) To obtain a credential, the subject needs a credential structure definition. This credential structure, which defines the attribute structure of the credential, is provided by the IdM Service, based on the attributes defined in Section 5.1.2 and managed by the Keyrock IdM. Once both parties have finished the initialization and they share the same credential definition, the issuer starts the Idemix proving protocol and computes a random value called *nonce* (to prove freshness) that is sent to the subject. Then, the subject computes a cryptographic message (also known as *crypto token*) according to the credential structure. The issuance message with the token is sent to the issuer, which creates a cryptographic part of the credential signing the attributes with his secret key. It also creates a proof of correctness. Furthermore, the issuer can save the pseudonym and the context for accountability purposes.

(4) Finally, the issuer replies to the subject by sending a cryptographic message with the proof of correctness and the attributes signature. The subject verifies this cryptographic material, and based on this message, it generates and stores the credential.

Once the subject has the Idemix credential, it can use such a credential to derive partial identities for a privacy-preserving authentication mechanism when accessing other IoT services.

5.2.3. M2M Claim-Based Authentication. The M2M authentication relies on the Idemix *proving* protocol, which is the main operation provided by the IdM system. Figure 4 shows the involved interactions that are required between a subject, which wants to prove the possession of certain attributes of its credential, when accessing a target's resource. Such interactions are explained as follows:

(1) The subject makes a request to an IoT service, which is hosted in a target entity (acting as a verifier). Then, this entity requires the subject to present a certain cryptographic proof to demonstrate the possession of a certain credential or specific identity attributes.

(2) The verifier computes a random value called *nonce* that is sent to the subject. Based on context, the *Identity Selector* component of the subject makes use of the *Credential Manager* module to select the most appropriate partial identity or pseudonym to be used against the verifier among the ones it already has available in its database. Optionally, if supported by the underlying cryptographic engine, in case the subject does know the proof specification that is required by the IoT service, the verifier can send a *presentation policy* stating which data must be disclosed by the subject to gain access to the requested IoT service. In other words, the presentation policy defines which credentials and attributes are required and/or which conditions have to be fulfilled by the attributes. The subject (acting as prover) defines the proof specification from the selected credential(s) to be used against the verifier. This proof includes the nonce and attributes, as well as statements about these attributes. Then, the prover builds a cryptographic object as proof and sends it along with the specification to the verifier.

(3) The verifier validates the incoming proof specification using the cryptographic proof. It computes the verifying protocols to check that the attributes statements and pseudonyms are valid. The verifier, depending on the validation result, can send an affirmative or negative response to the subject.

Furthermore, Idemix allows an entity to generate pseudonyms that demonstrate the possession of its master secret during the proving protocol explained above, enhancing unlikability. The proof generated by the subject proves to the target the knowledge of the secret associated with a pseudonym, as well as the fact that such a pseudonym is based on the subject's master secret key. That is, $nym \leftarrow g^{m_1}h^r$ where g and h are public common group parameters, m is the master secret key, and r is a randomization exponent. Then, the subject (as a prover) computes a challenge using a hash function, by considering the context and the computed proofs, that is, the CL proof and the pseudonym proof, as an input. Specifically, the prover computes random t-values of the form $t := g^r$, it get a challenge $c := H(\cdots \parallel t)$, and finally it computes a response s-values of the form $s := r - c\alpha$ to prove knowledge of α. Then, the target (as a verifier) computes the s-values that are hashed and compared against c. It should be noticed that although it is not required in our proposal, Idemix also allows proving more complex kind of proofs, which deal with inequalities and logic operators (AND, OR, and NOT) over the attributes contained in a credential.

Nonetheless, there are some cases in which the verifier requires from the prover the presentation of nonanonymizable attributes, such as national ID number, which undermines user's privacy against the verifier. In this particular case, the usage of pseudonyms is useless. In any case, it is up to the subject to consent this disclosure with a partial identity (Idemix proof) that reveals such attributes, ensuring the minimal disclosure principle.

5.3. Using the IdM System for Privacy-Preserving Credential Provisioning. The main goal of the proposed IdM system is to enable a privacy-preserving IdM approach for the IoT, in which users and smart objects can access to IoT services while their privacy is preserved. As an instantiation of an IoT service, the proposed IdM system has been used to allow an entity to obtain security credentials in a privacy-preserving way, by disclosing only a subset of its identity attributes. On the one hand, DCapBAC AuthZ tokens, which are generated from XACML decisions, can be requested by using Idemix to prove some identity attributes that are used for the authorization decision process. On the other hand, CP-ABE keys can be generated according to such proof and consequently associated with specific partial identities.

5.3.1. DCapBAC Integration for Access Control. Our IdM has been integrated with our capability-based access control DCapBac [9] that has been specially tailored for M2M interactions in IoT. In a basic scenario, the capability token can be directly validated by the target device that authenticates the subject with public key cryptography, without the intervention of the IdM Service. During the token generation process, the Authorization Service includes the subject's public key (associated with its X.509 certificate) in the token. Then, during the access request, the subject is forced to provide the token and its certificate to the target, which authenticates the subject and checks that its certificate's public key matches the one included in the token. This feature strengthens security, avoiding impersonation, which is one of the main disadvantages of bearer tokens that are currently employed by OAuth-based approaches.

Nonetheless, since the capability token contains the subject's public key, its privacy is not preserved. To cope with this issue, our IdM system allows users and smart objects to ask the Authorization Service for capability tokens in a privacy-preserving way, by providing its Idemix credentials instead of using their X.509 certificates. The Idemix *proving* process enables the subject to demonstrate that its identity is linked to the pseudonym included in the capability token. Figure 5 shows the three main stages required to carry out the anonymous authorization process based on the usage of the IdM system and DCapBAC.

In the first stage, the subject performs the *Idemix Credential Issuance* (messages (1)–(6)) protocol against the Idemix Issuer that was explained in Section 5.2.2. To this aim, the IdM Service authenticates the subject in the Keyrock IdM using the subject X.509 certificate. Once authenticated, the IdM verifies that the attributes included in the credential match the attributes held in Keyrock IdM for such an entity. If successful, the Idemix Issuer generates the cryptographic credential based on a particular structure that follows the attributes available in Keyrock. Notice that the issuance protocol requires a common understanding of certain group of parameters, as well as the credential specification structure, which are publicly available in the issuer.

Then, in the *AuthZ Token Request*, the subject performs the Idemix *proving* protocol against the Authorization Service, thereby demonstrating the necessary identity attributes required to get capability tokens in a privacy-preserving way. For this purpose, the verifier submodule of the Authorization Service verifies the cryptographic proof generated by the subject. The proof aims to prove the possession of a specific combination of identity attributes, along with the cryptographic proof of the pseudonym (steps (8)–(12)). Then, the *Token Manager* initiates the authorization stage to check whether the subject's attributes fulfill the XACML policy for a given action and resource. Thus, it performs a XACML request to the *Policy Decision Point* (PDP), which evaluates the policies and makes the authorization decision. If the PDP reports a *Permit* decision, the Token Manager generates the capability token, which includes the rights to access the target along with the pseudonym for the requesting subject. Although it is not shown in the diagram for the sake of clarity, at this stage, the authorization decision can be leveraged by our Trust-aware Access Control for IoT (TacIoT) [36] solution, which takes into account a multidimensional approach to quantify trust values about the subject reliability prior making the final access control decision.

Afterward, during the *access request* stage (message 18), the subject uses the capability token previously obtained to access to a particular resource hosted by the target. Thus, it generates an initial request with a session key SKsession, which, in turn, is encrypted with the public key PK of the target; that is,

$$\begin{aligned}&(18)\, Subject \longrightarrow Target\colon \big[EncryptPK\left(PK_{Target},\right.\\&\left.SK_{session}\right), EncryptSK\,(SK_{session},\\&(target \parallel action \parallel resource \parallel capToken))\big]. \end{aligned} \tag{1}$$

Then, the target gets the pseudonym included in the token and challenges the subject to prove, by means of Idemix, that it is the entity linked to the token. Namely, the subject starts the Idemix *proving* protocol to prove that it has a valid credential issued by the issuer, ensuring the data minimization principle. As in the Idemix *issuance* process, the proving protocol requires a common agreement between the parties, about certain system parameters. The cryptographic proof generated by the subject is derived from its credential and contains a pseudonym proof and the CL signature. Such a proof (and its specification) is sent (message (22)) to the target for verification. In case the pseudonym is valid and linked to the subject, it is authenticated, and therefore, the access is granted. This last interaction is, in turn,

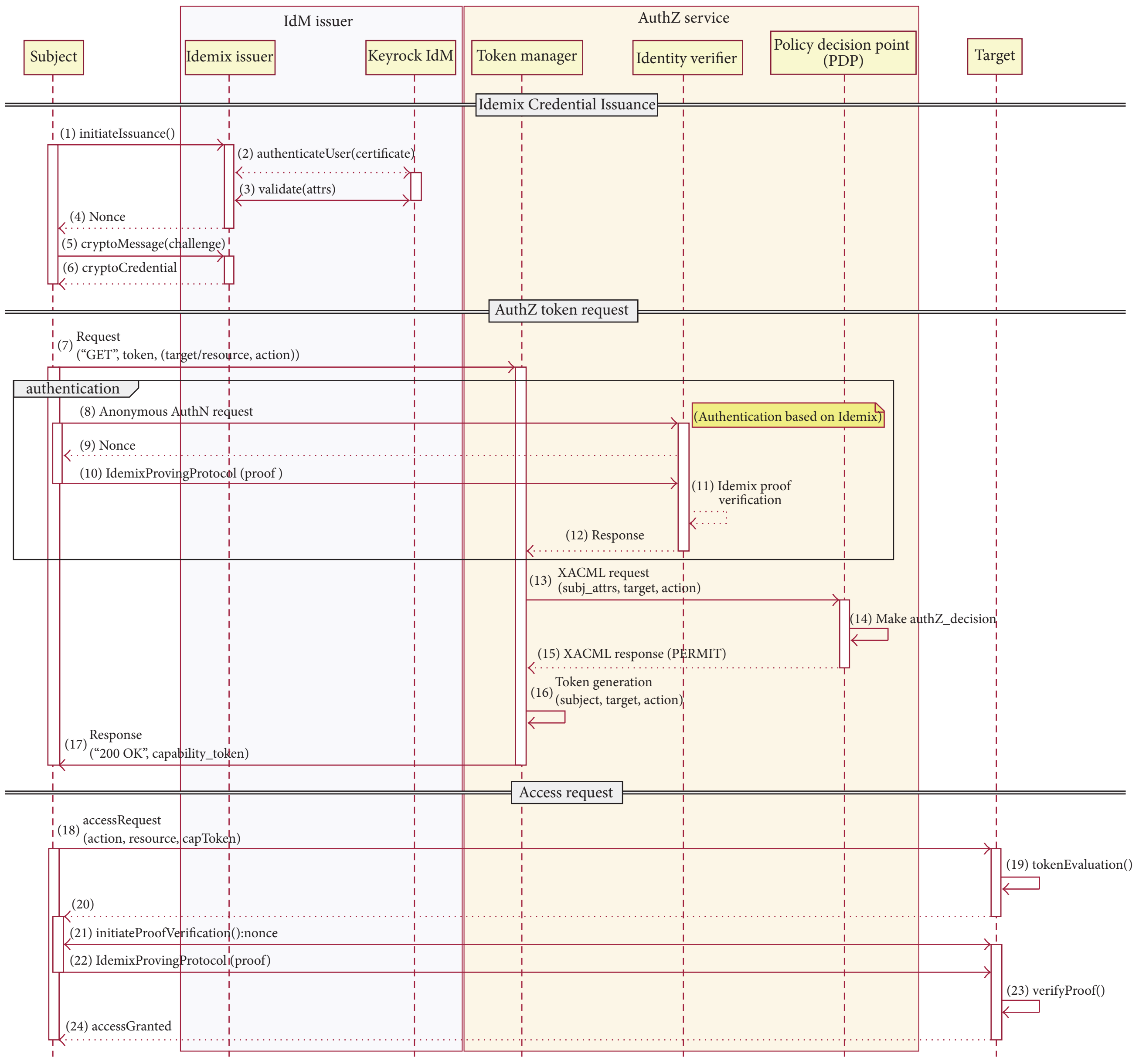

IdM issuer
AuthZ service
Subject
Idemix issuer
Keyrock IdM
Token manager
Identity verifier
Policy decision point (PDP)
Target
Idemix Credential Issuance
(1) initiateIssuance()
(2) authenticateUser(certificate)
(3) validate(attrs)
(4) Nonce
(5) cryptoMessage(challenge)
(6) cryptoCredential
AuthZ token request
(7) Request ("GET", token, (target/resource, action))
authentication
(8) Anonymous AuthN request
(Authentication based on Idemix)
(9) Nonce
(10) IdemixProvingProtocol (proof)
(11) Idemix proof verification
(12) Response
(13) XACML request (subj_attrs, target, action)
(14) Make authZ_decision
(15) XACML response (PERMIT)
(16) Token generation (subject, target, action)
(17) Response ("200 OK", capability_token)
Access request
(18) accessRequest (action, resource, capToken)
(19) tokenEvaluation()
(20)
(21) initiateProofVerification():nonce
(22) IdemixProvingProtocol (proof)
(23) verifyProof()
(24) accessGranted

Figure 5: IdM and access control.

encrypted with the session key previously established among the parties.

$$(24)\, \textit{Target} \longrightarrow \textit{Subject}\colon\ \left[\textit{EncryptSK}\left(SK_{session}, \textit{resourcevalue—target}\right)\right]. \quad (2)$$

5.3.2. CP-ABE Integration for Confidential Data Outsourcing. Following a similar approach, our IdM system has been integrated with the CP-ABE scheme, so users and smart objects are enabled to apply for CP-ABE keys in a privacy-preserving way. In this way, the proposed IdM system allows entities to demonstrate their attributes against the Key Manager Service in order to obtain necessary cryptographic material associated with the attributes. Under CP-ABE scheme [11], a piece of data can be encrypted under a policy of attributes such that only those target entities satisfying the policy (and therefore holding the associated identity attributes and keys) are able to decrypt the shared information. In our approach, CP-ABE keys are managed by a *Trusted Third Party* (TTP), called *key manager* (KEM). A basic CP-ABE scenario includes three main entities: a data *producer*, one or more data consumers, and the Key Manager Service that is responsible for generating the appropriate keys for all participants. The role of these components is determined by their participation in the algorithms of a CP-ABE scheme [11], which defines four main protocols:

(i) Setup (Key Manager Service): the algorithm generates the public parameters, which are common to all users

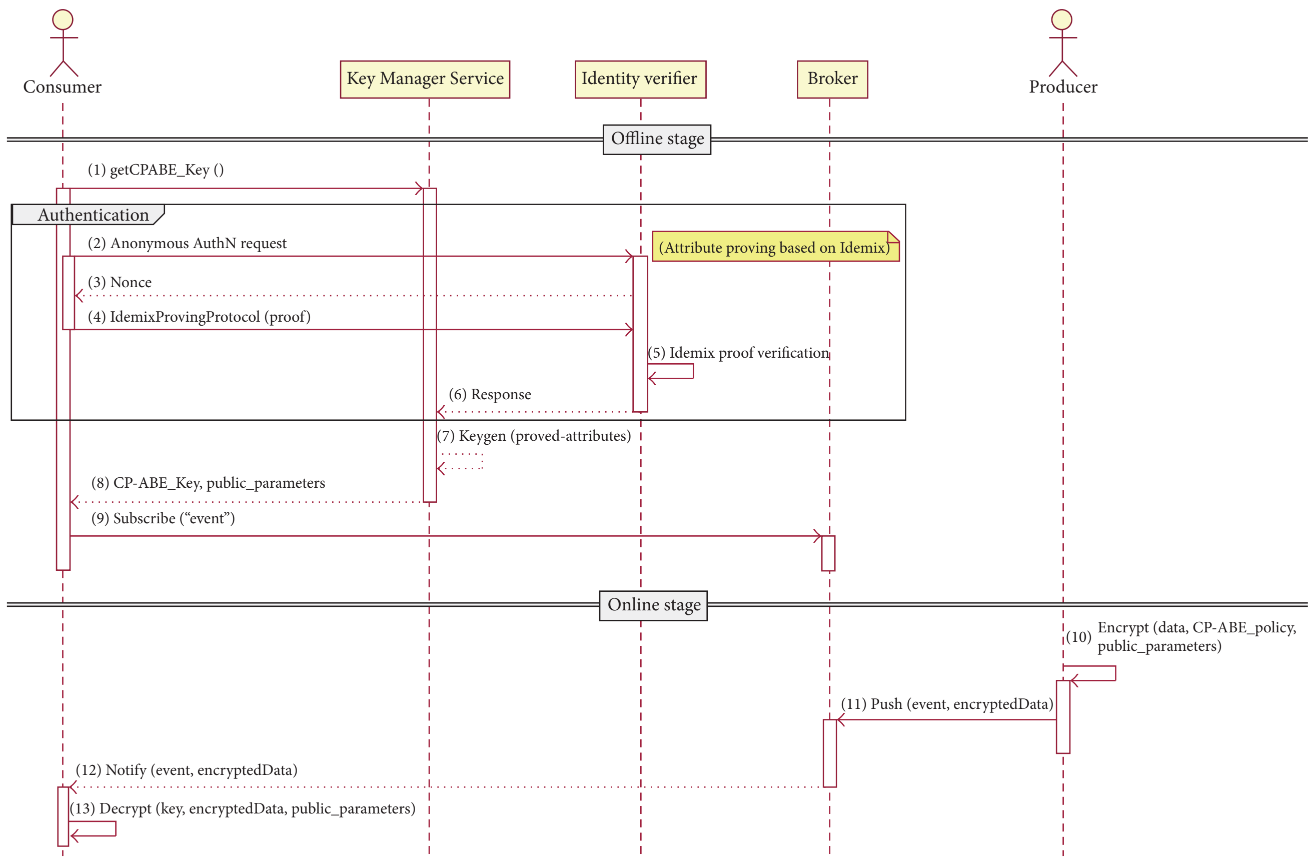

Consumer
Key Manager Service
Identity verifier
Broker
Producer
Offline stage
(1) getCPABE_Key ()
Authentication
(2) Anonymous AuthN request
(Attribute proving based on Idemix)
(3) Nonce
(4) IdemixProvingProtocol (proof)
(5) Idemix proof verification
(6) Response
(7) Keygen (proved-attributes)
(8) CP-ABE_Key, public_parameters
(9) Subscribe ("event")
Online stage
(10) Encrypt (data, CP-ABE_policy, public_parameters)
(11) Push (event, encryptedData)
(12) Notify (event, encryptedData)
(13) Decrypt (key, encryptedData, public_parameters)

FIGURE 6: IdM in an Attribute-Based Encryption scenario.

of the system, as well as a master secret key that is used to generate CP-ABE keys.

(ii) Key Generation (Key Manager Service): after an entity proves that it has a certain set of attributes by using the Idemix protocol, the algorithm takes as input the master key and these attributes. The result is a CP-ABE key for decryption.

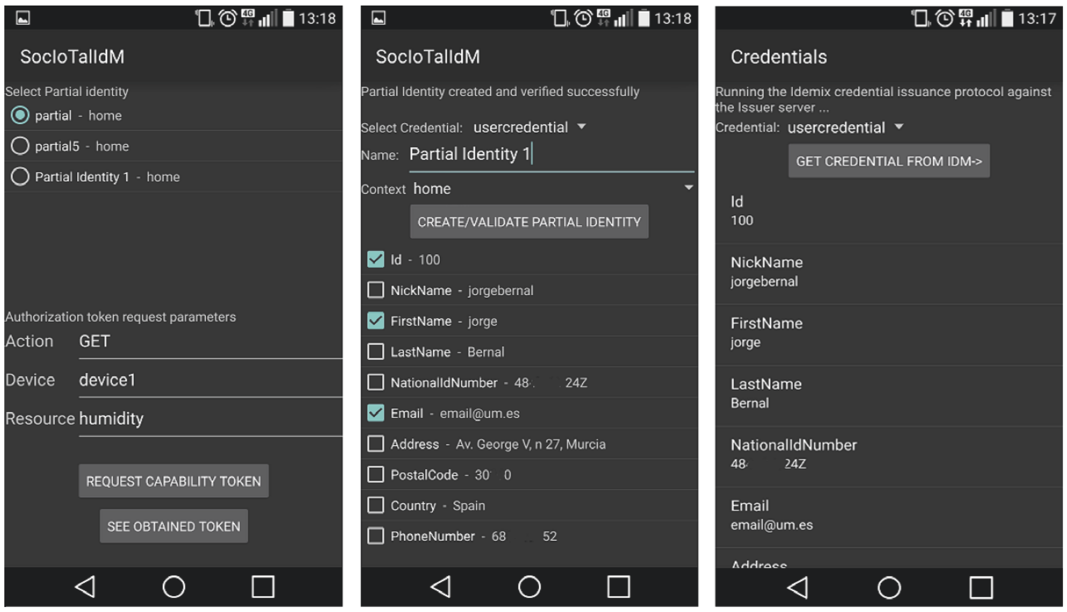
(iii) Encrypt (producer): it takes the message, public parameters and an encryption policy representing a combination of attributes that must be satisfied to decrypt the message. The result of this algorithm is a ciphertext containing such policy.

(iv) Decrypt (consumer): the decryption algorithm takes the public parameters, the ciphertext, and a CP-ABE key as an input. If the attribute set (associated with the CP-ABE key) satisfies the policy, the consumer will be able to decrypt the ciphertext.

Figure 6 depicts a publish/subscribe scenario for IoT based on CP-ABE. As, in addition to smart objects acting as producers and consumers, as well as the Key Manager Service, this communication model is based on the use of a *broker* in charge of receiving data from producers to be published to consumers. The broker functionality may be provided by another smart object or deployed on a central data platform.

Firstly, during an *offline stage*, the involved producers and consumers entities obtain the cryptographic material required for a secure information exchange based on CP-ABE. To this end, they authenticate themselves against the Key Manager Services (acting as an Idemix Verifier) and demonstrate their attributes by using the Idemix *proving* protocol (messages (2)–(6)). Then, the Key Manager Service distributes the CP-ABE public parameters (common for all participants), as well as the corresponding CP-ABE private key, which is associated with the proved attributes (messages (7)-(8)). After the initialization, subscriber entities perform the subscription to a certain topic (message (9)).

Secondly, during an *online stage*, a producer decides to publish certain information, by sending a message that consisting of a ciphertext that is the result of the execution of *encrypt* algorithm (message (10)-(11)). Thus, only those entities (i.e., a group) with CP-ABE keys satisfying the CP-ABE policy used to encrypt data will be able to decrypt the information (message (13)). Notice that the broker entity cannot decrypt the information unless it is endowed also with the proper keys. In addition, the broker is a transparent 3rd party that cannot trace user's behavior as users do not need to be authenticated against the broker, avoiding linkability. The authentication-authorization is done directly by the fact of being able to decrypt the data. And the key manager is only consulted once to get the keys.



13:18
SocIoTalIdM
Select Partial identity
● partial - home
○ partial5 - home
○ Partial Identity 1 - home
Authorization token request parameters
Action GET
Device device1
Resource humidity
REQUEST CAPABILITY TOKEN
SEE OBTAINED TOKEN

13:18
SocIoTalIdM
Partial Identity created and verified successfully
Select Credential: usercredential
Name: Partial Identity 1
Context home
CREATE/VALIDATE PARTIAL IDENTITY
✓ Id - 100
☐ NickName - jorgebernal
✓ FirstName - jorge
☐ LastName - Bernal
☐ NationalIdNumber - 48 24Z
✓ Email - email@um.es
☐ Address - Av. George V, n 27, Murcia
☐ PostalCode - 30 0
☐ Country - Spain
☐ PhoneNumber - 68 52

13:17
Credentials
Running the Idemix credential issuance protocol against the Issuer server ...
Credential: usercredential
GET CREDENTIAL FROM IDM->
Id
100
NickName
jorgebernal
FirstName
jorge
LastName
Bernal
NationalIdNumber
48 24Z
Email
email@um.es
Address

Figure 7: Screenshots of the IdM Android app.

6. Implementation and Performance

After having described the actors and interactions related to the proposed IdM system, this section aims to provide a set of experimental results associated with the main processes of our approach.

6.1. Implementation. Our IdM system is composed of seven main components, which have been integrated and evaluated as part of our testbed. Below, we provide a description of each component and its relationship with entities that are defined in the previous section.

(i) IdM-Android-Client: it is a new implemented Android app that allows obtaining Idemix credentials from the issuer server. It interacts with the verifier, which can validate the partial identity (Idemix proof) derived from the Idemix credential. A user-friendly interface is given to the user to select the attributes that she wants to add to each partial identity. The implementation uses the already existing Idemix Java library (Idemix version 2.3.0) ported from Java to Android by [41]. It is worth mentioning that the testbed uses Android SDK 1.7 and the RSA secret key length is established to 1024 bits (i.e., 80-bits security level). Some screen shots of our app are depicted in Figure 7. It also allows obtaining a capability token from the Authorization Service using the partial identity.

(ii) Issuer Server: it is a web application implemented with Java servlets and XML-RPC that allows generating Idemix credentials for clients. The credentials are generated according to the user (or smart object) profile in Keyrock. Communications are done by HTTPS. The client must be authenticated against the issuer using a valid certificate or he can be authenticated by the Keyrock server.

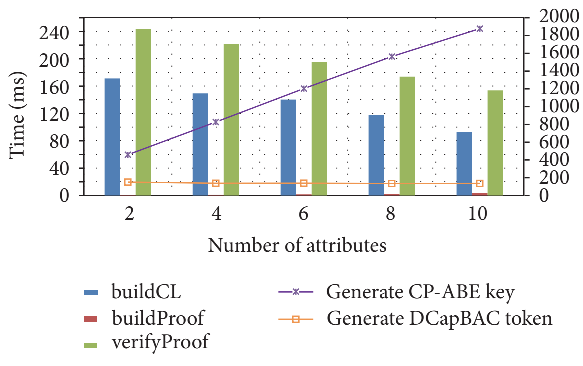
(iii) Verifier-Server: it is a web application, also implemented with Java servlets and XML-RPC, which is able to validate partial identities (implementing verifier functionality) presented by the client application.

(iv) Authorization Service: it is a web application that allows users to obtain capability tokens using their partial identities. In other words, it allows authenticating and demonstrating their attributes by means of Idemix proofs of having a valid credential issued by the issuer. Additionally, this entity acts as a HTTP client requesting authorization decisions to the PDP service. It also plays the role of SocIoTal-Verifier-Server. Furthermore, the service also incorporates a web-based PAP service, where users are enabled to define access control policies for those smart objects.

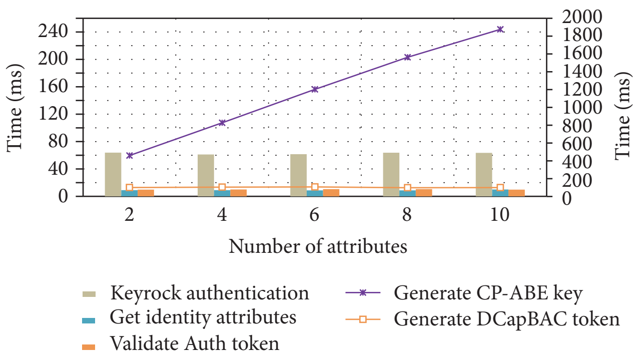
(v) IdM Keyrock Client API: the SocIoTal IdM Keyrock Client Java library provides a basic API for identity management by implementing a client to interact with the Keyrock server. To carry out such communication, the SCIM 2.0 and Identity API v3 interfaces provided by this IdM system are used.

(vi) Keyrock server: this is a modified deployment of the FIWARE Keyrock IdM system. In particular, the Keyrock server has been extended to also cope with smart objects' identities, in addition to users. It has been also extended to implement attributes of the SCIM standard [8] that was not included in Keyrock.

(vii) Web User Environment: it is a front-end dashboard that integrates various server side SocIoTal components (as client APIs) and provides a user-friendly GUI for end users. It interacts with Keyrock



(a) Idemix-based credential provisioning



(b) Keyrock-based credential provisioning

FIGURE 8: Testbed performance.

and allows registering users and managing their attributes.

(viii) Key Manager Service: it is a HTTPS server implemented in Java servlets accepting requests for CP-ABE keys generation. These keys, as already mentioned, are associated with the attributes stored in the Keyrock server. It also plays the role of SocIoTal-Verifier-Server. A client library has been also developed to interact with the Key Manager Service.

The aforementioned software modules are open source, and they are available in the SocIoTal repository in GitHub (https://github.com/sociotal), where a wiki with a complete documentation of the software is also provided.

6.2. Performance Evaluation. The testbed consists on evaluating the performance of the IdM system when a subject tries to get DCapBAC tokens and CP-ABE keys (Section 5.3) by using different authentication methods. Specifically, the subject makes use of the Android application to get such credentials through classical password-based authentication (Section 5.2.1), as well as the privacy-preserving Idemix-based approach (Section 5.2.2).

Figure 8 shows two graphs that sum up the performance obtained in the testbed. The charts show the time required by the subject to obtain DCapBac tokens and CP-ABE keys. On the one hand, Figure 8(a) shows the time required when the privacy-preserving authentication method is employed (i.e., using the Idemix-enabled Android application). On the other hand, Figure 8(b) shows the time required to obtain the credentials, but relying on the traditional authentication method (i.e., by means of basic online authentication using our API that interacts with the Keyrock server).

The x-axis in both charts represents the number of attributes that are disclosed to the target, that is, the Authorization Service that generates the DCapBAC token or the Key Manager Service that generates the CP-ABE keys. In the left chart, the Idemix authentication operations are depicted by the vertical bars with y-axis in the left, while the times for the generation of DCapBAC token and CP-ABE keys are depicted in line charts with y-axis on the right. Notice that in this case the time scale is different, since the credentials generation takes more time.

The first series identifies the time required by the Android app to build the Idemix CL proof *buildCL*, which is the heaviest task in the proving protocol. Secondly, the *verifyProof* operation shows the time required by the server side (either Authorization Service or Key Manager Service) to verify the Idemix cryptographic proof, including the time required to verify CL signature.

Notice that, in the Idemix authentication case, all the proofs contain the 10 attributes to comply with the credential structure, and the different lies in the amount of those 10 attributes disclosed. The time required to build and validate such a proof depends on the number of attributes revealed. As more attributes are revealed, that is, not encrypted in the proof, Idemix library requires less time to handle it.

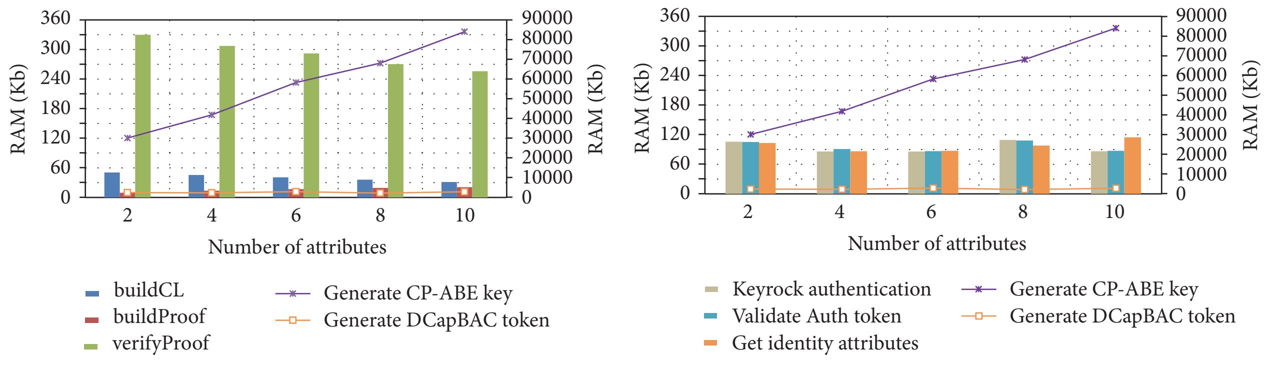
As it can be seen in Figure 8(b), the traditional authentication is faster than Idemix authentication in Figure 8(a). Despite the fact that traditional authentication requires an additional request to Keyrock server to obtain the user attributes, the whole time required to perform the authentication plus obtaining the required attributes plus validating the AuthZ token is less than the time required in the proving protocol carried out through Idemix. Notice that the traditional authentication (in Figure 8(b)) is fully delegated to the server (where are taken the times), while in Idemix case the cryptography operations of *builCL* and *buildProof* are performed in the device. In both charts, the operations "Generate CP-ABE Key" and "Generate DCapBAC Token" are performed in the server.

The times required to obtain the authorization decision and generate the capability token are usually steady across the different tests. It should be noted that the results do not include request or network delays. Besides, the time required to generate the DCapBAC token includes two main tasks: (1) the time required by the PDP to make the decision—around 65% of the time—and (2) the time required to build up the token.

In addition, the testbed has evaluated the memory consumption required by each of the processes described previously. The memory results are shown in Figure 9. Again, two different y-axes are used in the figures as the

Table 1: Identity management systems comparison.

Feature	Solution/technology							
	SAML	OpenID	OAuth	Shibboleth	U-Prove	Idemix	Keyrock	Our IdM
Confidentiality and integrity	++	++	++	++	++	++	++	++
Single Sign-On	++	++	++	++	+	+	++	++
Transparency	++	++	++	++	+	+	++	+
Strong authentication	+	+	+	+	++	++	+	++
Federation support	+	+	+	++	–	–	++	+
Intervention	–	++	++	++	++	++	++	++
Minimal disclosure information	–	++	+	+	++	++	+	++
Usability	++	++	++	++	+	+	++	+
Attribute revocation	++	++	+	++	+	+	+	+
Nonreputation	++	++	++	++	++	++	++	++
Offline M2M authentication	–	–	–	–	++	++	–	++
Pseudonymity	+	+	+	+	++	++	+	++
Unlikability	–	–	–	+	+	++	–	++
Zero-knowledge proofs	–	–	–	–	++	++	–	++
Attribute aggregation	–	–	–	+	+	+	–	+



(a) Memory consumption in Idemix-based credential provisioning (b) Memory consumption in Keyrock-based credential provisioning

Figure 9: Memory consumption.

credentials generation operations depicted in line charts required considerably more memory to accomplish their task. As it can be seen, the memory consumption trends behave in most of the cases according to computation times evaluated in Figure 8. In addition, the obtained performance results follow a linear evolution trend, which indicates the feasibility of the proposal.

7. Security Analysis and IdMs Comparison

This section provides a security analysis as well as a comparison of our IdM system against the most relevant IdM technologies in the current state of the art. The comparison analyzes different features that an ideal IdM solution should meet and compares them with our proposal. Most of these features have been taken from previous IdM analysis such as [42, 43]. Table 1 summarizes this comparative, which analyzes SAML, OpenID, OAuth, U-Prove, Idemix, and Keyrock IdM against the IdM system presented in this paper. In the table, the symbol + means that the feature is partially satisfied by the analyzed IdM technology, the symbol ++ means that the feature is fully supported, and the symbol – indicates that such a functionality is not supported.

An IdM system must guarantee that the information is shared and accessible only by appropriate entities, securing communications to ensure that confidentiality and integrity principles are met. This feature is fulfilled by all the analyzed IdM solutions. Nonetheless, in SAML, OpenID, and OAuth the user is redirected to his home domain for authentication, so that there is a higher risk of impersonation in case the user is redirected to a malicious IdP. Besides, these solutions do not directly support user control of identity data, meaning that the IdP is responsible for dealing with this aspect. Moreover, Idemix and U-Prove allow end users to manage their attributes by themselves without IdP intervention.

Single Sign-On (SSO) property allows entities to have a unique account when accessing across different service providers, by using a single authentication assertion for the session. Although Anonymous Credential Systems (ACS), such as Idemix and U-Prove, do not make use of these assertions, this feature can be easily achieved thanks to the usage of presentation protocol processes. Nonetheless, they are less efficient since they require a higher computation time to evaluate the cryptographic proofs for each service request. Usability property is also only partially fulfilled by ACS, which usually requires advanced skills for its usage and maintenance.

The IdM system should provide transparency so that privacy-relevant data processing can be understood and reconstructed anytime. In this sense mechanisms for logging and auditing are needed in order to trace event occurring in the system, ensuring that any entity cannot deny having accomplished any action. In this regard, in SAML, OpenID, OAuth, and Keyrock, the IdP server can easily log the transactions and end users access. However, in a pure ACS like Idemix, the auditing feature is more difficult to fulfill because of their unlinkability properties. In our IdM system, auditing can be performed when the access is done using traditional authentication through Keyrock, but the feature cannot be easily achieved when the entity interacts directly with the target entity by means of Idemix.

Nowadays, authentication based on shared secrets (e.g., username-password) is being replaced by stronger security mechanisms, such as authentication based on digital certificates and biometric or attribute-based credentials. In SAML, OpenID, OAuth, and Keyrock, the authentication is left to the IdP, whereas ACS already provide strong authentication per se, by linking attributes to cryptographic keys.

SAML, OpenID, and OAuth can be used to enable identity federation so that different IdP can establish trust between each other. Indeed, Shibboleth exploits SAML to provide a complete federated identity management solution. Keystone, Keyrock, and in turn our IdM support this feature as they can be configured to install and work with Shibboleth. However, federation capabilities in ACS are still not properly addressed yet.

Intervenability is defined as "the property that intervention is possible concerning all ongoing or planned privacy-relevant data processing." It is implemented by mechanisms such as the end user consent feature, which is achieved by most of the analyzed solutions (except SAML). It can be done either by redirecting and asking users to the IdP or by allowing users to specify the particular set of attributes employed in a partial identity, as it is done in ACS. Indeed, end users in ACS are given full control of their identity data, while in traditional IdMs, the IdP is in charge of managing the identity information.

Minimal disclosure of information refers to the ability to selectively reveal as less information as possible in the credentials presented to a target service. U-Prove, Idemix, and our IdM system support this feature since a user or smart objects is enabled to select which particular attributes want to disclose. In traditional IdM systems, such as OpenID, the minimal disclosure feature can be achieved due to the user consent functionality. However, in SAML, this functionality cannot be easily achieved since, once authenticated, the SP is able to request any attribute allowed by the IdP.

However, ACS are currently difficult to manage, as final users are supposed to have advanced technical knowledge in order to manage their identities and interact with users. In this sense, our IdM system provides user-friendly apps and tools over Idemix (as can be shown in Figure 7), which aid users to manage their partial identities against different IoT services.

Users might lose their right to carry a particular credential, or their attributes may change over time, so that the IdM system should be able to revoke user's existing attributes. At the same time, SPs should be able to check, at anytime, whether a subject's attribute is still valid or not. This feature is easily achieved in traditional IdM systems, since the IdP (or Attribute Provider) maintaining the attributes can stop releasing or validating a revoked attribute or credential. However, in ACS, the user might try to use invalid credential after revocation, so that a third revocation entity is needed. For instance, in Idemix the revocation entity publishes a white-listing that is employed by a revocation scheme based on accumulators. In addition, Idemix also supports short-term claims meaning that they have to be renovated so often.

The nonreputation property guarantees that an end user or entity cannot deny having made an action. As the assertions can be digitally signed and the anonymous credentials are also tight to the users this security principle is met by all the IdM solutions analyzed.

One of the advantages provided by ACS and our IdM system and not supported by traditional IdM approaches is that they allow performing offline M2M authentication. It can be done since the verifier can validate the cryptographic proof without the need of an online TTP in charge of verifying the correctness and validity of the assertions. This feature is of utmost importance in M2M IoT scenarios, where devices are not necessarily connected to the Internet.

In SAML, the issued assertions allow using pseudonyms to preserve the end user's privacy. In other words, target accessed entities do not know the end user's real identity. Nonetheless, the IdP could trace the user's access, since the IdP has to generate a different assertion each time they need to access to the target. In contrast, Idemix and, in turn, our IdM system support unlinkable pseudonyms. Unlinkability is defined as "the property that privacy-relevant data cannot be linked across domains that are constituted by a common purpose and context." In Idemix, pseudonyms are scoped-exclusive, so that it can generate a cryptographically unique pseudonym for given "scope string," limiting, for instance, the number of pseudonyms for a particular domain.

Zero-knowledge proof means that the user can create a cryptographic proof to convince the verifier that he has or "knows" a valid credential obtained from the issuer, while the user can repeat such a proof as many time as he wants without linking the individuals proofs and without revealing the secret information. ACS allow making claims about an attribute without revealing the attribute and prove inequalities; for instance, Idemix allows proving that the value of an attribute is within a certain range or certain attributes have the same value without revealing such a value.

The user can prove the possession of different credentials obtained from different issuers in one single proof, meaning that different attributes can be aggregated in one partial identity. This attribute aggregation feature is not supported by traditional IdM systems. Nonetheless, more sophisticated IdMs approaches, such as [44], are also able to support this feature without relying on cryptographic approach, but at the expense of introducing an online TTP called *Identity Aggregator*, which is in charge of holding, aggregating, and managing the user's attributes coming from different IdPs.

In conclusion, despite the fact that some features can be only partially supported, such as transparency, usability, and attribute revocation-aggregation, our IdM system supports most of the security and privacy features analyzed in this section, which can be considered the desirable ones in a holistic IdM system.

8. Conclusions and Future Work

This paper has presented an IdM system which provides a novel holistic and privacy-preserving IdM solution to deal with heterogeneous IoT scenarios that require both traditional online access control and authentication, along with a claim-based approach for M2M interactions. The IdM system relies on the Idemix anonymous credential system for claim-based authentication, as well as on FIWARE IdM system (Keyrock) to deal with classic IdM aspects. The paper has shown how the IdM system allows obtaining different kind of security credentials (DCapBAC tokens and CP-ABE keys) in a privacy-preserving way. Idemix is used in our IdM system to derive cryptographic proofs, thereby allowing minimal disclosure of identity attributes when obtaining those security credentials, which are used later for accessing IoT services. As future work, we expect to develop and deploy our IdM system also in constrained IoT devices, allowing managing the M2M authorization process between IoT constrained devices using the Idemix proving protocol.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work has been partially funded by the European Project "ReliAble euRopean Identity EcoSystem (ARIES)" from European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement no. 700085 and also by the Spanish MICINN, project EDISON (TIN2014-52099-R).

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] Z. M. Fadlullah, M. M. Fouda, N. Kato, A. Takeuchi, N. Iwasaki, and Y. Nozaki, "Toward intelligent machine-to-machine communications in smart grid," *IEEE Communications Magazine*, vol. 49, no. 4, pp. 60–65, 2011.

[3] G. Kortuem, F. Kawsar, V. Sundramoorthy, and D. Fitton, "Smart objects as building blocks for the internet of things," *IEEE Internet Computing*, vol. 14, no. 1, pp. 44–51, 2010.

[4] B. Krishnamurthy and C. E. Wills, "On the leakage of personally identifiable information via online social networks," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Online Social Networks, WOSN '09*, pp. 7–12, 2009.

[5] M. Langheinrich, "Privacy by design-principles of privacy-aware ubiquitous systems," in *Proceedings of the Ubicomp 2001: Ubiquitous Computing*, Lecture Notes in Computer Science, pp. 273–291, Springer, Berlin, Germany, 2001.

[6] D. Recordon and D. Reed, "OpenID 2.0: a platform for user-centric identity management," in *Proceedings of the 2d ACM Workshop on Digital Identity Management, DIM '06. Co-located with the 13th ACM Conference on Computer and Communications Security, CCS '06*, pp. 11–16, November 2006.

[7] J. Camenisch and E. V. Herreweghen, "Design and implementation of the idemix anonymous credential system," in *Proceedings of the 9th ACM Conference on Computer and Communications Security CCS '02*, pp. 21–30, ACM, New York, NY, USA, November 2002.

[8] P. Hunt, K. Grizzle, E. Wahlstroem, and C. Mortimore, "System for Cross-domain Identity Management: Core Schema," RFC Editor RFC7643, 2015.

[9] J. L. H. Ramos, M. P. Pawlowski, A. J. Jara, A. F. Skarmeta, and L. Ladid, "Toward a lightweight authentication and authorization framework for smart objects," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 4, pp. 690–702, 2015.

[10] E. Rissanen, "extensible access control markup language (xacml) version 3.0 oasis standard," 2012.

[11] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of the IEEE Symposium on Security and Privacy (SP '07)*, pp. 321–334, IEEE, New York, NY, USA, May 2007.

[12] J. L. Hernández-Ramos, J. B. Bernabé, and A. Skarmeta, "ARMY: architecture for a secure and privacy-aware lifecycle of smart objects in the internet of my things," *IEEE Communications Magazine*, vol. 54, no. 9, pp. 28–35, 2016.

[13] A. Bassi, M. Bauer, M. Fiedler et al., *Enabling Things to Talk*, Springer, Berlin, Germany, 2013.

[14] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266–2279, 2013.

[15] Z. Yan, P. Zhang, and A. V. Vasilakos, "A survey on trust management for Internet of Things," *Journal of Network and Computer Applications*, vol. 42, pp. 120–134, 2014.

[16] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: the road ahead," *Computer Networks*, vol. 76, pp. 146–164, 2015.

[17] J. Hughes and E. Maler, "Security Assertion Markup Language (SAML) v2. 0 technical overview," OASIS SSTC Working Draft sstc-saml-tech-overview-2.0-draft-08, pp. 29–38, 2005.

[18] A. C. Sarma and J. Girão, "Identities in the future internet of things," *Wireless Personal Communications*, vol. 49, no. 3, pp. 353–363, 2009.

[19] J. M. Such, A. Espinosa, A. Garcia-Fornes, and V. Botti, "Partial identities as a foundation for trust and reputation," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 7, pp. 1128–1136, 2011.

[20] J. Camenisch and A. Lysyanskaya, "An efficient system for non-transferable anonymous credentials with optional anonymity revocation," in *Advances in cryptology-EUROCRYPT 2001*, pp. 93–118, Springer, Berlin, Germany.

[21] C. Paquin and G. Zaverucha, "U-prove cryptographic specification v1.1," Tech. Rep., Microsoft, New Mexico, NM, USA, 2011.

[22] A. Sabouri, I. Krontiris, and K. Rannenberg, "Attribute-based credentials for trust (ABC4Trust)," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7449, pp. 218-219, 2012.

[23] T. Heer, O. Garcia-Morchon, R. Hummen, S. L. Keoh, S. S. Kumar, and K. Wehrle, "Security challenges in the IP-based Internet of Things," *Wireless Personal Communications*, vol. 61, no. 3, pp. 527–542, 2011.

[24] D. Gessner, A. Olivereau, A. S. Segura, and A. Serbanati, "Trustworthy infrastructure services for a secure and privacy-respecting internet of things," in *Proceedings of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom '12*, pp. 998–1003, June 2012.

[25] D. Wang and P. Wang, "On the anonymity of two-factor authentication schemes for wireless sensor networks: Attacks, principle and solutions," *Computer Networks*, vol. 73, pp. 41–57, 2014.

[26] A. Alcaide, E. Palomar, J. Montero-Castillo, and A. Ribagorda, "Anonymous authentication for privacy-preserving IoT target-driven applications," *Computers & Security*, vol. 37, pp. 111–123, 2013.

[27] L. Seitz, G. Selander, and C. Gehrmann, "Authorization framework for the Internet-of-Things," in *Proceedings of the 2013 IEEE 14th International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM '13*, pp. 1–6, June 2013.

[28] D. Hardt, "The OAuth 2.0 Authorization Framework," RFC Editor RFC6749, 2012.

[29] R. L. Morgan, S. Cantor, S. Carmody, W. Hoehn, and K. Klingenstein, "Federated security: the shibboleth approach," *Educause Quarterly*, vol. 27, no. 4, pp. 12–17, 2004.

[30] J. Camenisch and A. Lysyanskaya, "A Signature Scheme with Efficient Protocols," in *Security in Communication Networks*, Lecture Notes in Computer Science, pp. 268–289, Springer, Berlin, Germany, 2003.

[31] M. Zorzi, A. Gluhak, S. Lange, and A. Bassi, "From today's INTRAnet of things to a future INTERnet of things: a wireless- and mobility-related view," *IEEE Wireless Communications*, vol. 17, no. 6, pp. 44–51, 2010.

[32] S. Sun, L. Lannom, and B. Boesch, "Handle System Overview," RFC Editor RFC3650, 2003.

[33] E. Rescorla, "HTTP Over TLS," RFC Editor RFC2818, 2000.

[34] E. Rescorla and N. Modadugu, RFC 6347: Datagram transport layer security (DTLS), Request for Comments, IETF, 2012.

[35] Z. Shelby, RFC 6690: Constrained RESTful Environments (CoRE) Link Format, Request for Comments, IETF, 2012.

[36] J. Bernal Bernabe, J. L. Hernandez Ramos, and A. F. Skarmeta Gomez, "TACIoT: multidimensional trust-aware access control system for the Internet of Things," *Soft Computing*, vol. 20, no. 5, pp. 1763–1779, 2016.

[37] Z. Shelby, K. Hartke, and C. Bormann, "The Constrained Application Protocol (CoAP)," RFC Editor RFC7252, 2014.

[38] M. Nottingham, Web linking, Internet-Draft draft-nottingham-rfc5988bis-06, IETF Secretariat, June 2017.

[39] R. Housley, W. Polk, W. Ford, and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," RFC Editor RFC3280, 2002.

[40] IBM Research Zurich, "Specification of the identity mixer cryptographic library," Tech. Rep., 2013.

[41] M. Heupel, "Porting and evaluating the performance of idemix and tor anonymity on modern smart-phones," 2010.

[42] E. Birrell and F. B. Schneider, "Federated identity management systems: a privacy-based characterization," *IEEE Security and Privacy*, vol. 11, no. 5, pp. 36–48, 2013.

[43] G. Dólera Tormo, F. Gómez Mármol, and G. Martínez Pérez, *Identity Management in Cloud Systems*, Springer, Berlin, Germany, 2014.

[44] A. Pérez, G. López, O. Cánovas, and A. F. Gómez-Skarmeta, "Formal description of the SWIFT identity management framework," *Future Generation Computer Systems*, vol. 27, no. 8, pp. 1113–1123, 2011.