Anonymous certification for E-assessment opinion polls

Nesrine Kaaniche[1] · Chistophe Kiennert[2] · Maryline Laurent[2] · Joaquin Garcia-Alfaro[2]

Abstract
Anonymous certification (AC) refers to cryptographic mechanisms in which users get certified from trusted issuers, with regard to some pre-defined user attributes, in order to produce presentation tokens. Such tokens satisfy service providers' access policies, without revealing sensitive user information. AC systems are generally classified under two main different categories: (1) one-time show credentials that can be shown once for avoiding their originating user being traced from one transaction to another, and (2) multi-show credentials that can be used many times while avoiding their originating user to be traced. In this paper, we consider e-assessment opinion polls scenarios and propose an AC scheme where the one-time show property is relevant for making sure each user cannot hand in more than one poll in order to get significant results. To mitigate cheating, the scheme is provided with two extra procedures: attribute revocation and anonymity removal. The cor-rectness of our scheme, as well as unforgeability, privacy and anonymity removal, are analyzed and demonstrated.

Keywords Security and protection · Access control · Management of computing and information systems · Privacy · Anonymous credentials · Attribute-based signatures · Bilinear pairings · Anonymous certification

1 Introduction

Anonymous certification (AC), also known by privacy preserving certification, allows users to prove they are authorized to access a resource without revealing more than they need about their identity. For example, users can be issued with certified attributes that may be required by the system verifier, such as *Older than 18*, *works at IBM*, or *lives in the UK*. When the users want to prove that they own the right set of attributes, they perform a digital signature based on the required attributes, allowing the system verifier to check if a precise user is authorized, sometimes without even knowing precisely which attributes were used.

In this paper, we explore the integration of AC under e-assessment services, i.e., on-line services expected to evaluate learners' tasks. AC can be integrated in e-assessment services, whenever it is not necessary to fully identify the learner. For example, when learners need to access a given course material on a virtual learning environment [e.g., Moodle by Cole and Foster (2007)], it should be enough to prove that the learner comes from an allowed university and registered for the course. That way, it becomes impossible for the learning environment to track the activities of learners, while still granting access to the learners to the course material.

When a learner takes an assessment, the learner's work can be anonymously sent to anti-cheating tools (such as anti-plagiarism). With AC, each tool might receive a request for the same work without being able to know which learner wrote it, but also without being able to correlate the requests and decide whether they were issued by the same learner. Under this context, we present e-$\mathcal{PCS}$, a privacy preserving certification scheme that is being integrated under the scope of TeSLA(TeSLA Consortium 2016), a H2020 EU-funded project that aims at providing learners with innovative authentication and authorship environments. e-$\mathcal{PCS}$

✉ Joaquin Garcia-Alfaro
garcia_a@telecom-sudparis.eu

Nesrine Kaaniche
n.kaaniche@sheffield.ac.uk

Chistophe Kiennert
christophe.kiennert@telecom-sudparis.eu

Maryline Laurent
maryline.laurent@telecom-sudparis.eu

Department of Computer Science, University of Sheffield, Sheffield, England, UK

Institut Polytechnique de Paris, Telecom Sud-Paris, SAMOVAR CNRS UMR 5157, Paris, France

builds over the attribute-based signature scheme of previous constructions by Kaaniche et al. (2017) and Kaaniche and Laurent (2016).

The solution presented in this paper is an extension of previous work by Kaaniche et al. (2017). As explained below, the extension details formal threat models and security analysis, emphasizes the support of one-time show unlinkability property and introduces a *proof-of-concept* of the proposed framework. The contributions of this work are as follows:

1. e-$\mathcal{PCS}$ allows a user to show his presentation policy, based on his certified credentials, for only once in each different poll. However, the same credential can still be shown anonymously in another event without being linked. e-$\mathcal{PCS}$ also permits a user to prove the possession of a credential, with regards to a presentation policy in as many polls as necessary.
2. e-$\mathcal{PCS}$ introduces the inspection procedure, relying on a trusted third party that aims to remove the anonymity of e-learners, in case of—*concerns*—reported by the university/verifiers. For this purpose, two main algorithms have been added namely Trace and Judge, in order to genuinely conduct the inspection process and provide a proof of judgment.
3. we provide formal system and security models for e-$\mathcal{PCS}$ framework. For instance, we discuss the resistance of e-$\mathcal{PCS}$ against two adversaries, relying on two different threat models. We prove that our proposed scheme satisfies the confidentiality, the unforgeability, the privacy and the anonymity removal requirements.
4. we provide an informal detailed discussion related to prospective supported features, to enhance the applicability of e-assessment opinion polls in different settings. Main functional requirements include the support of multiple issuing authorities and credentials' revocation processing.

The e-$\mathcal{PCS}$ scheme has several advantages. Firstly, it does not rely on a trusted third party (TTP) to protect users' privacy. It inherits the privacy preservation property from the anonymous certification procedures. Secondly, it is a resource-saving mechanism as it does not rely on an interactive protocol for obtaining certified credentials, thanks to the use of attribute-based signatures. Thirdly, e-$\mathcal{PCS}$ does not leak any information on who has participated to the poll and who has not, and e-learners among different opinion polls are unlinkable w.r.t. different and independent verifiers. Finally, in order to prevent malicious actions and to mitigate to anonymity abuse, e-$\mathcal{PCS}$ relies on a trusted inspection authority responsible for revoking the anonymity of an originating user when needed.

Paper[1] *organization* Sections 2 and 3 provide the background and related work, elaborating further on AC and the TeSLAarchitecture. Section 4 presents our contributions. Section 5 discusses the resistance of our scheme against security and privacy attacks. Section 6 concludes the paper.

2 Anonymous certification (AC)

Privacy preserving certification, often referred to as privacy preserving attribute based credentials (AC), was first presented by Chaum (1985), to protect users' privacy in transactions' systems. Later, this promoting idea has been formalized by Camenisch and Lysyanskaya (2001). Since then, different concrete constructions have been proposed and considered as essential elementary units in privacy-preserving identity-management systems. In fact, each honest user is able to prove to a requesting service provider, that he holds some authenticated attributes, known also as credentials, obtained from authorized and trusted issuing authorities.

2.1 Definitions

AC systems rely on some well identified entities. As shown in Fig. 1, three main entities are considered as *mandatory* namely the issuer, the verifier and the user, while both a revocation authority and the tracing authority are *optional*. Indeed, in an AC system, each user (i.e., TeSLA e-learner) represents a pivotal entity, who aspires a privacy preserving access to requested services, afforded by service providers, referred to as verifiers (i.e., TeSLA cloud domain). Each verifier imposes an access control policy, called *presentation policy*, to its resources and services, while enforcing a set of credentials that have to be owned by the users. To do so, each user has first to obtain credentials from a trusted issuing authority, known as issuer (i.e., institution domain). Then, he selects the appropriate information (i.e., a subset of certified attributes) from the credentials and shows the selected information to the requesting service provider, under a *presentation token*. To effectively generate and accurately verify presentation tokens, the most recent revocation information has to be gathered from the revocation authority (i.e., service within the institution domain), by the user, respectively the verifier. That is, the revocation authority is responsible for revoking issued credentials and maintaining a list of valid credentials. In case of revocation, revoked credentials will not be longer allowed to derive tokens. Tracing authorities

[1] This paper is an extended and revised version of a former conference work by Kaaniche et al. (2017).

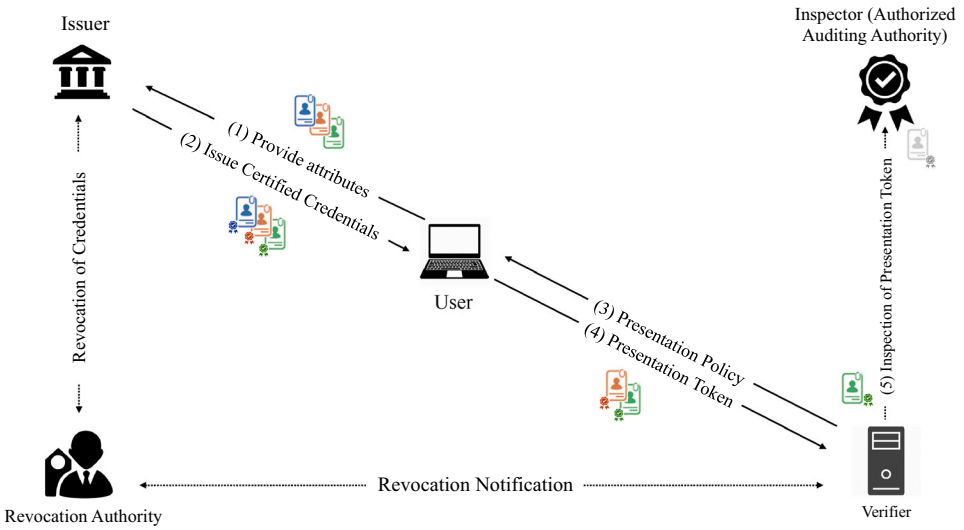

Fig. 1 AC entities and procedures

refer to auditing entities trusted by the system, which can effectively handle user anonymity, if requested.

Unlinkability and untraceability are the most desired privacy features a system should support. However, there are several cases, when they can lead to misuse and where anonymity removal is necessary. This type of feature is called inspection. It is the responsibility of the dedicated entity referred to as tracing authority to trace a presentation token when needed. The presentation policy must specify the tracing authority identity (i.e., tracing authority's public key) and which information the tracing authority must be able to recover. The user creates the presentation token which contains encrypted versions of the requested attribute values with the required public key of the tracing. Additionally, this prover provides a verifiable cryptographic proof that the encrypted content contains the same attribute values as encoded in the certified credential.

2.2 Related work

Privacy-preserving authentication mechanisms mainly rely on the use of malleable signatures' schemes and zero-knowledge proofs of knowledge protocols. Indeed, to transform a credential, i.e., signed attributes, into a presentation token, the user mainly has to create a zero-knowledge proof showing that he possesses a valid signature on a committed value over his attributes, received from an authentic issuing organisation. Interested readers may refer to Lindell and Katz (2014) for more details about cryptographic primitives, namely zero-knowledge proofs and commitments schemes. Well-known examples encompass the signature scheme by Brands (2000), mainly inspired from blind signatures, and the signature scheme by Camenisch and Lysyanskaya (2001), essentially relying on the concept of group signatures, which have been implemented in Microsoft U-Prove by Paquin and Zaverucha (2011) and by Camenisch et al. (2010); IBM (2018), respectively.

Attribute-based signatures (ABS) are considered as a promoting cryptographic primitive for building privacy-preserving authentication scheme, as suggested by Maji et al. (2011). Each user possessing a set of attributes, first obtains a secret signing key per attribute, generated by a trusted central entity, referred to as attribute authority. The user can then sign, e.g., a document, w.r.t. a predicate, i.e., access policy, satisfied by the set of attributes he holds. Considering different design approaches, several ABS constructions have appeared in literature. ABS schemes by Li et al. (2010), Belguith et al. (2017) and Herranz et al. (2012) propose the requirement of satisfying access structures under threshold policies. Schemes by Maji et al. (2011) and Zhang and Feng (2012) propose the use of monotonic policies, while Okamoto and Takashima (2011) propose the use of non-monotonic policies. With regard to the distribution of keys, Shahandashti and Safavi-Naini (2009), Maji et al. (2011) and Zhang and Feng (2012) propose single authority distribution; vs. multiple authority distribution by Okamoto and Takashima (2011) and Belguith et al. (2018a). Related literature also exists on extended schemes supporting practical features, such as attributes' revocation and hidden access policies (Xu et al. 2018; Liu et al. 2017; Belguith et al. 2018b).

Kaaniche and Laurent (2016) propose a complete privacy-preserving authentication system, called $\mathcal{HABS}$, is introduced. The proposed protocol is built over the use of a novel ABS construction. It is designed considering the following features: (1) signature traceability, permitting to grant authorized auditing entities the ability of identifying the user originating a given ABS-message couple; (2) unlinkability between different issuing entities, to mitigate against colluding ABS authorities trying to link user

requests; and (3) mitigation of replayed sessions, via secure timestamping. The approach, extended by Kaaniche et al. (2017) and Kiennert et al. (2017a), is the foundational scheme of the contribution presented in this paper (i.e., e-$\mathcal{PCS}$). With regards to e-$\mathcal{PCS}$, previous work does not support one-time unlinkability property. The new approach also supports inspection features that comply with the EU General Data Protection Regulation (EU 2016/679 GDPR). GDPR introduces the new obligation of accountability for organizations, such that each entity processing personal data must be able to provide compliance of auditing authorities.

E-learning systems are presented by Aïmeur et al. (2008) and Aïmeur and Hage (2010) as a composition of Internet-based protocols and advanced security tools. These tools are mainly a set of cryptographic mechanisms, that allow learners to perform on-line studies while protecting their privacy. Aïmeur et al. also survey a list of security and challenges that have to be addressed, pointing out common threats that may harm the privacy of learners. Solutions such as attribute-based encryption and anonymous certification are listed as future work in their conclusions.

Work by Gathuri et al. (2014) raises the problem of impersonation issues in e-assessment applications. The authors proposed to combine profile-based authentication scheme with time-stamping techniques to avoid impersonation issues. Even though the proposed technique is efficient in terms of computation and communication overheads, it does not fulfill privacy requirements to assessed students. Kim and Huh (2018) study existing e-learning systems that are widely used in universities and educational institutions and suggested ways of improving these systems' performance and structural problems with a view to developing novel interactive and secure plateforms. Wu and Wu (2019) suggest a new criteria evaluation scheme, based on candidates' attributes identified from multiple data entries, and verified by a multi-level selection process.

3 The TeSLAproject

The TeSLA project is a EU-funded project. It addresses e-assessment challenges. E-assessments are at the center of novel online education sectors. The goal of the project is to offer e-learners to take remote assessments, e.g., to avoid mandatory attendance constraints. It must provide equivalent guarantees to the learners, with respect to traditional examination scenarios in face-to-face situations. Addressing physical attendance constraints paves the way for significant cost-effective learning and assessment approaches. Current achievements of the project to-date are of technological nature, such as its modular, secure and privacy-preserving design that integrates authentication and authorship verification of learners.

From a security and privacy standpoint, the main properties ensured by TeSLA are authentication and authorship. Authentication aims at proving an entity's identity to another party; authorship consists in proving the identity of the creator of a piece of work. Some other traditional properties, in terms of confidentiality and integrity must be assured as well. In Kiennert et al. (2017b), some security and privacy aspects of the TeSLAe-assessment system were analyzed and discussed. In turn, the TeSLAplatform was designed to comply the EU General Data Protection Regulation (GDPR). In terms of learners' certification techniques, previous work by Kaaniche and Laurent (2016) and Kiennert et al. (2017a) highlights the necessity of enforcing privacy-preserving attribute credentials, i.e., to ensure that service verifiers authenticate learners in an anonymous manner.

Privacy preserving certification schemes are powerful cryptographic mechanisms that provide data minimization cryptographic schemes, permitting users to reveal only required information to service providers. These schemes rely on some organizations issuing certificates for each user's attributes in a way that users can demonstrate possession of attributes in a series of transactions without being linked. This property, referred to as multi-show property, has first been formerly presented by Camenisch and Lysyanskaya (2001), allowing a user to unlinkably prove possession of a credential as many times as necessary. However, in some applications, this reusability property is too flexible to be useful, namely for e-voting, electronic surveys, etc.

3.1 The TeSLAarchitecture

The TeSLAarchitecture, from a global perspective (cf. Fig. 2a) is comprised of several components that may belong to two domains: a cloud-computing domain (shared among several institutions) and the institution system domain (one per university). Components that belong to the institution domain (cf. Fig. 2b) must execute at the local infrastructure of each university willing to make use of the TeSLA e-assessment framework, while components that belong to the cloud domain are completely independent of the university resources. The two domains do not share data unless explicitly stated.

The institution domain may contain the following components: (a) the TeSLAE-assessment portal (TEP), which acts as a service broker that gathers and forwards requests to other TeSLAcomponents; (b) the reporting tool (RT), that aims at gathering statistics regarding the e-assessment activities; (c) the TeSLAinstruments, which handle authentication and authorship data from and toward the client side.

The institution domain also interacts with already existing *virtual learning environment* (VLEs), which can be provided by using classic *learning management systems* (LMS) such as Moodle (cf. https://moodle.org/). A plugin integrated

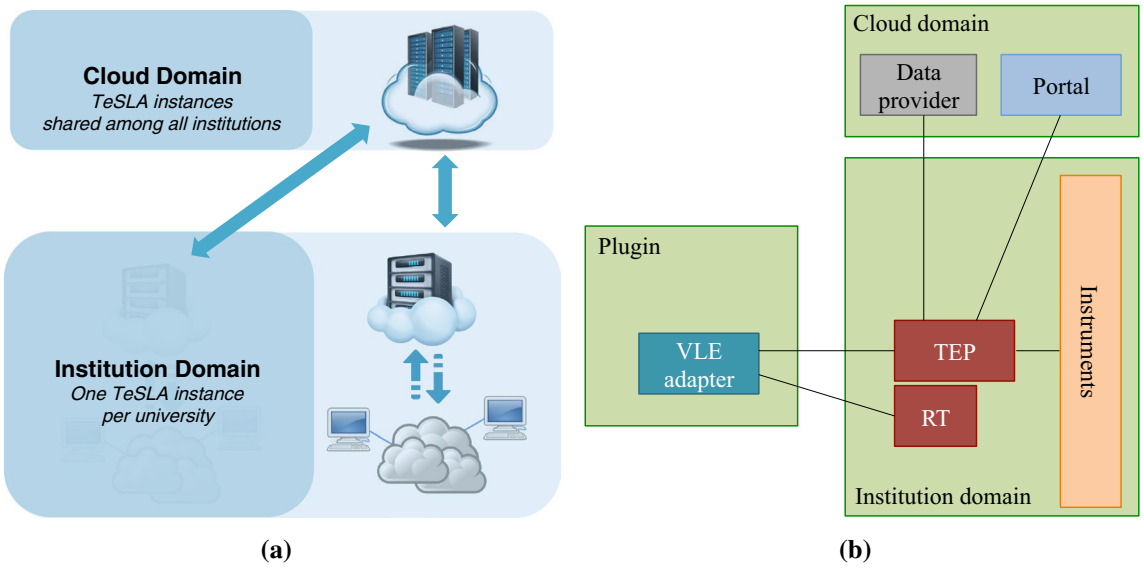

Fig. 2 The TeSLAarchitecture. **a** Global perspective. **b** Main entities and components, in which anonymous certification (AC) is placed as one of the instruments depicted in the figure (cf. online implementation at https://github.com/jgalfaro/mirrored-PKIPCS)

to the VLE acts as a client side interface with the TeSLA components.

Some other tools that require integration to the VLE, potentially executed at the cloud domain, for outsourcing and performance reasons, may need to send requests and data to the TeSLAcomponents at the institution domain through the same plugin. This includes learner tools, instructor tools, and external tools. The learner tool and the instructor tool are respectively designed to take or setup e-assessments. External tools are in charge of handling authentication and authorship data and sending them back to TeSLAinstruments for evaluation, e.g., in terms of the anti-cheating countermeasures. In addition, a TeSLAIdentity Provider (TIP), which is in charge of handling identity details related with learners, will finally conduct an identity mapping that is used with all the other TeSLAcomponents. The communication between all the components is secured by the TLS protocol [cf. Rescorla and Dierks (2008)], deployed on the whole architecture with mutual authentication, hence ensuring confidentiality and integrity of every data exchange. The underlying public key infrastructure for TLS deployment and management is fully detailed in Kiennert et al. (2017b).

3.2 Security and functional requirements

During the execution of e-assessment opinion polls, learners are requested to fill up several forms, including multiple-choice answers or fill-in-the-blank fields. Anonymity shall be enforced while requesting access to the forms. Authorized user must participate only once, i.e., the system must ensure double participation of learners, in order to have significant results. This anonymity-accountability trade-off has to be resolved while defining an efficient privacy-preserving certification mechanism that fulfills the following security and privacy properties. We assume the following requirements:

- *Completeness and soundness* Completeness means that all valid access requests (originated from an authorized e-learner) should be counted correctly, while soundness implies all invalid access requests (double access requests) should not be counted.
- *Anonymity* The e-learner must remain anonymous during the access request process.
- *Unforgeability* An unauthorized e-learner should not be able to provide a valid proof of certified credentials to authenticate with the service provider.
- *Issue-show unlinkability* To preserve users' privacy against colluding malicious entities, this property ensures that it is unfeasible to link any information gathered during the credential issuance phase to its corresponding user while running the presentation phase.
- *One-time show unlinkability* This property prevents any e-learner to respond to a specific poll twice.
- *Accountability* To ensure accountability and prevent anonymity abuse, it is necessary to identify the originating user. This feature has to be carried out by a trusted tracing authority.
- *Low computation and communication costs* The proposed scheme should offer acceptable computation and communication costs, mainly for resource-constrained resources.

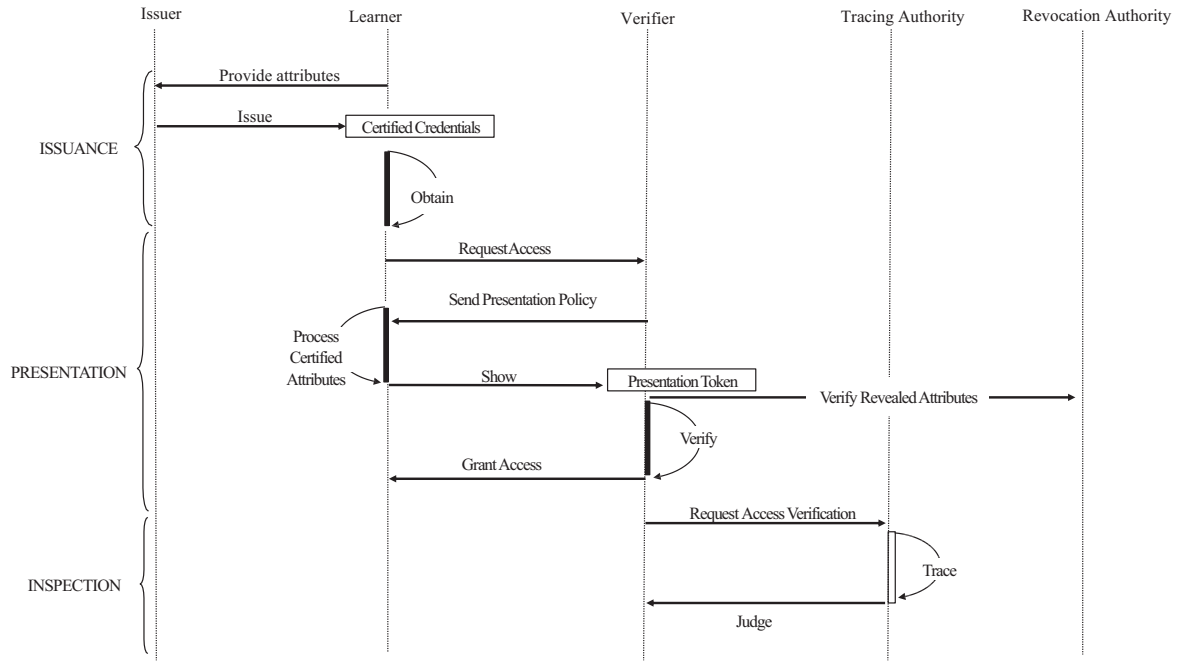

Fig. 3 Workflow associated to e-$\mathcal{PCS}$

4 The e-$\mathcal{PCS}$ construction

We move now to presenting e-$\mathcal{PCS}$, as a privacy-preserving authentication scheme integrated to the TeSLA framework as one of the authentication instruments of the architecture. The solution extends the existing attribute-based signature scheme by Kaaniche et al. (2017). It incorporates a novel traceability feature using presentation tokens. The new procedure extends the original construction by relaxing anonymity when the identification of presentation token user is needed. Nevertheless, and to prevent an issuing organisation to trace the users, the extended construction incorporates a tracing authority, i.e., a tracing authority (cf., Fig. 1). Several presentation tokens should be mapped to unique credentials, for statistical or pricing requirements. The $\mathcal{PCS}$ construction in Kaaniche et al. (2017) is extended and adapted to support the aforementioned features. As such, e-$\mathcal{PCS}$ extends $\mathcal{PCS}$ with four new procedures, namely: Initialization, Issuance, Presentation and Inspection.

4.1 Overview

e-$\mathcal{PCS}$ relies on four procedures and eight randomized algorithms. During the Initialization phase, two algorithms, namely the Setup and KeyGen are executed by a central trusted entity to set-up the system and generate all entities public and private keys, respectively. Figure 3 shows the different interactions between the system entities, and points out the main procedures and algorithms:

The Issuance phase is an interactive protocol, between the user (i.e., the learner) and the issuer (i.e., the institution domain). It involves two algorithms, called Issue and Obtain, run by the issuer and the user respectively. At the end of this phase, the issuer provides a credential to the user, certifying the validity of the contained attributes. Recall that each user may have several credentials, each asserting some collection of attributes. The new phase (i.e., Presentation) relies on the requests of a user to get granted access to the resources of the service provider (i.e., the get access to the resources of TeSLA at the Cloud domain). During this two-party interactive phase, the service provider first sends to the requesting user the presentation policy.

It defines which proofs have to be provided, and which information from the credential(s) have to be revealed. To do so, the user checks the set of credentials that may fulfill the access policy in order to generate the presentation token. The verifier then checks the correctness of the received token based on public parameters provided by the issuing organisation(s).

This interactive phase includes two algorithms, Show and Verify. Show is executed by the user. Verify is executed by the service provider, which must create and send a blinded group element, denoted as $\mathtt{M}$, and which is based on a random value m. The blinded group element is sent to the requesting user. Then, the user signs the blinded group

element by using his certified credentials, relying on some selected attributes that satisfy the presentation policy of the service provider.

The last phase relies on the execution of the INSPECTION procedure, which is carried by a separate and trusted entity, referred to as the auditing authority. The procedure relies on the execution of two algorithms, Trace and Judge, which are required to identify the user and give a valid proof of judgment.

4.2 System model

e-$\mathcal{PCS}$ relies on eight randomized algorithms defined as follows:

Setup—given the security parameter ξ, this algorithm generates the global public parameters *params* and a pair of public and private keys (pk_t, sk_t) for the tracing authority.

KeyGen—given the public parameters *params*, the KeyGen algorithm derives the pair of public and secret keys for both users as well as the issuing organization (s). The public and secret keys are noted respectively (pk_{u_j}, sk_{u_j}) for user j and (pk_o, sk_o) for the issuing organization. Hereafter, we assume that the public parameters also include the public key of the tracing authority, and all the algorithms have *params* as a default input.

Issue—executed by the issuing authority, the Issue algorithm takes as input the public key of the user pk_u, a set of attributes $\mathcal{S} \subset \mathbb{S}$ (where $\mathcal{S} = \{a_i\}_{i=1}^{N}$, N corresponds to the number of attributes and $\mathbb{S}$ represents the attribute universe), the private key of the issuing organization sk_o and the public key of the tracing authority pk_t. It outputs a credential C associated to the set of attributes $\mathcal{S}$.

Obtain—the user runs the Obtain algorithm to check the consistency of the received credentials. This algorithm takes as input the credential C, the private key of the user sk_u, the public key of the issuing organization pk_o and eventually the public key of the tracing authority pk_t. It outputs a bit $b \in \{0, 1\}$, referring to the validity or invalidity of the credentials received from the issuing entity.

Show—the user executes the Show algorithm to generate the proof of possession of some attributes. As input, the Show algorithm gets a nonce (i.e., the randomized message $\mathtt{M}$), the signing predicate Υ, the private key of the user sk_u, the user credential C and a series of attributes $\mathcal{S}\simeq$, i.e., subset of $\mathcal{S}\simeq$, such as $\Upsilon(\mathcal{S}\simeq) = 1$. The output of the Show algorithm is the Σ signature (or an error message $\perp$).

Verify—performed by the service provider. The algorithm received as inputs m, Σ, pk_o (the public key of the issuing organization(s)), and the signing predicate Υ. The binary output of the algorithm is either 0 or 1, where 1 refers to *acceptance* of the given signature-message; 0 corresponds to *rejection*.

Trace—executed by the tracing authority, the Trace algorithm receives as inputs sk_t (the secret key of the tracing authority), pk_o (the issuing public keys of the organizations) and Σ (the signature). The Trace algorithm provides as outputs the index j, which denotes user who signed $\mathtt{M}$, regarding the proof of judgement (ϖ) and the signing policy (Υ).

Judge—it receives the following inputs: pk_o (public keys of the issuing organizations), j (user index), Σ (signature), and ϖ (proof of judgement). The binary output of the algorithm is either 0 or 1, where 1 refers to a valid ϖ (i.e., user j is genuinely the originating entity behind Σ); 0 denotes the opposite.

4.3 Construction composition

In terms of composition, we can now refine the previous algorithms as follows:

- The Setup algorithm, which is in charge of generating the asymmetric bilinear group environments $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e})$, receives as input ξ (the security parameter). $\hat{e}$ represents an asymmetric bilinear function defined as $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$. The public key of the tracing authority is the couple defined as $pk_t = (h_1, h_2)$. The private key of the tracing authority is $sk_t = \alpha$. Let $\mathcal{H}$ be a cryptographic hash function, the remaining global parameters of the system are defined next:

  $$params = \{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, p, g_1, \{\gamma_i\}_{i\in[1,U]}, g_2, pk_t, \mathcal{H}\}$$

  where $g_1, h_1 = {g_1}^{\alpha}, \{\gamma_i\}_{i\in[1,U]} \in \mathbb{G}_1$ and $g_2, h_2 = {g_2}^{\alpha} \in \mathbb{G}_2$ represent the random generators built during the Setup algorithm, where the $\alpha \in \mathbb{Z}_p$ and U define the maximum number of attributes supported by the system (cf. Karchmer and Wigderson (1993) for details). Values in γ_i are involved in the construction of the secrets associated to the attributes a_i.
- The KeyGen algorithm, which is in charge of providing the key pairs (i.e., private and public keys) to the users and the issuing organizations. Each user obtains a pair of keys (sk_u, pk_u) where pk_u is the couple defined as $pk_u = (X_u, Y_u) = ({h_1}^{sk_u}, \hat{e}(g_1, g_2)^{sk_u})$ is the public key and sk_u, a randomly selected integer, is the private key of the user. The issuing organization obtains the private and public keys defined as (sk_o, pk_o), where $sk_o = (s_o, x_o) = (s_o, {g_1}^{s_o})$ (i.e; s_o is a randomly selected integer) and $pk_o = (X_o, Y_o) = (\hat{e}(g_1, g_2)^{s_o}, {h_2}^{s_o})$.
- The Issue algorithm is performed by the issuing organization to derive the user credential, w.r.t. the pre-shared set of attributes $\mathcal{S} \subset \mathbb{S}$. More specifically, $\mathbb{S} = \{a_1, a_2, \cdots, a_N\}$ represents the attribute universe, where N is the number of attributes. Given the public key of the user pk_u, the secret key of the issuing entity

sk_o, and the set of attributes $\mathcal{S}$, then the Issue algorithm returns the user credential as follows:

$$C = (C_1, C_2, \{C_{3,i}\}_{i\in[1,N]}, C_4)$$
$$= (x_o \cdot [X_u{}^{s_o \mathcal{H}(\mathcal{S})^{-1}}] \cdot h_1{}^{r}, g_2{}^{r}, \{\gamma_i{}^{r}\}_{i\in[1,N]}, g_1{}^{-r})$$

where r is a randomly chosen integer, $\gamma_i{}^{r}$ is the secret key associated to the attribute a_i and $\mathcal{H}(\mathcal{S}) = \mathcal{H}(a_1)\mathcal{H}(a_2)\cdots\mathcal{H}(a_N)$.

- The Obtain algorithm is performed by a user that wants to verify the consistency of the received credential. The inputs associated to the Obtain algorithm are the following: C (the credential), sk_u (private key of the user), pk_o (public key of the issuing organization) and $\mathcal{S}$ (the set of attributes). Equation (1) represents the verification process associated to the Obtain algorithm:

$$\hat{e}(C_1, g_2) \stackrel{?}{=} X_o \cdot \hat{e}(g_1{}^{sk_u \mathcal{H}(\mathcal{S})^{-1}}, Y_o) \cdot \hat{e}(h_1, C_2) \quad (1)$$

- The Show and Verify algorithms are associated to the Presentation phase, which can be seen as a two-party protocol, as follows:

Algorithm 1 Show algorithm

Input: $params$ (public parameters), $\mathcal{S}$ (set of attributes), C (credential), $\mathtt{M}$ (message), Υ (such that $\Upsilon(\mathcal{S}) = 1$ and the secret key of the user sk_u)
Output: $\Sigma = (\Omega, \sigma_1, \sigma_2, C'_1, C'_2, C'_4, A, \mathcal{S}_R)$
Convert Υ to a Linear Secret Sharing Schemes (LSSS) access structure (M, ρ) (cf. Beimel (1996)), such that M is constructed as a $l \times k$ matrix, whose rows are mapped to the attributes in accordance to an injective function ρ;
Compute $\mathbf{v} = (v_1, \cdots, v_l)$; such that $\mathbf{v}M = (1, 0, \cdots, 0)$;
$r' \xleftarrow{R} \mathbb{Z}_p{}^*$;
$C'_1 \leftarrow C_1 \cdot h_1{}^{r'} = x_o \cdot [X_u{}^{s_o \mathcal{H}(\mathcal{S})^{-1}}] \cdot h_1{}^{r+r'}$;
$C'_2 \leftarrow C_2 \cdot g_2{}^{r'} = g_2{}^{r+r'}$;
$C'_4 \leftarrow C_4 \cdot g_1{}^{-r'} = g_1{}^{-(r+r')}$;
$\Omega \leftarrow \{\}$;
$\sigma_1 \leftarrow C'_1$;
$C'_3 \leftarrow \{\}$;
for all $i \in \{1, \cdots, l\}$ **do**
  $\gamma'_i \leftarrow \gamma_i{}^{r} \cdot \gamma_i{}^{r'}$;
  $C'_3 \leftarrow C'_3 \cup \{\gamma'_i\}$;
  $\sigma_1 \leftarrow \sigma_1 \cdot \gamma'_{\rho(i)}{}^{v_i}$;
  $\omega_i \leftarrow C'_2{}^{v_i}$;
  $\Omega \leftarrow \Omega \cup \omega_i$;
end for
$r_m \xleftarrow{R} \mathbb{Z}_p$;
$A \leftarrow Y_o{}^{\frac{sk_u}{r_m}}$;
for all $a_i \in \mathcal{S}_H$ **do**
  $A \leftarrow A^{\mathcal{H}(a_i)^{-1}}$;
end for
$\sigma_2 \leftarrow g_1{}^{r_m}$;
$\sigma_1 \leftarrow \sigma_1 \cdot \mathtt{M}^{r_m}$;
return $\Sigma = (\Omega, \sigma_1, \sigma_2, C'_1, C'_2, C'_4, A, \mathcal{S}_R)$

1. When the service provider receives a request from a user to get access to a resource, the service provider executes the first phase of the Verify algorithm, in order to define the presentation policy. The presentation policy includes $\mathtt{M} = g_1{}^{m}$ (the randomized message), Υ (the access structure) and $\mathcal{S}$ (the set of attributes that have to be revealed). Some requirements must be taken into account. First, and to protect the construction from replay attacks, the value of m is unique to every authentication session. Second, the m value is shared by all the entities (e.g., users participating in the poll). The set of attributes must equal the set of attributes revealed to the verifier, i.e., $\mathcal{S}_R$, together with the set of attributed non-revealed to the verifier, i.e., such that $\mathcal{S} = \mathcal{S}_R \cup \mathcal{S}_H$.
2. The requesting user executes the Show algorithm, receiveing as main input the secret key of the user (sk_u), the user credential (C), the set of attributes $\mathcal{S}$ associated to both the user credential and the public key of the user (pk_u), the randomized message $\mathtt{M}$ and the access structure (Υ). The complete process is executed as detailed in Algorithm 1. The process returns the presentation token, defined as follows:

$$\Sigma = (\Omega, \sigma_1, \sigma_2, C'_1, C'_2, C'_4, A, \mathcal{S}_R)$$

where $\Omega = \{\omega_1, \cdots, \omega_l\}$ is the set of committed element values of vector $\mathbf{v}$ underlying Σ, i.e., the signature of $\mathtt{M}$ under Υ and credential's items.
3. Upon reception of Σ (i.e., the presentation token), the Verify algorithm receives as input parameters pk_o (the public key of the issuing entity), $\mathcal{S}_R$ (the set of revealed attributes), m (the message) and Υ (the signing policy). Based on those previous parameters, the verifier checks the one-time show property. Indeed, we note that the verifier keeps a local database logging all the participating entities to a poll, as depicted Table 1. That is, for each participating user, the verifier saves the received presentation tokens and calculates the following two values:

$$T_a = \hat{e}(C'_1, g_2) \quad T_b = \hat{e}(h_1, C'_2)$$

Before checking the received signature Σ_j from a user U_j, the verifier checks the validity of Eq. (2):

$$\forall i \in [1, q], \quad T_a{}^{(*)} \cdot T_b{}^{(i)} \stackrel{?}{=} T_a{}^{(i)} \cdot T_b{}^{(*)} \quad (2)$$

where q is the number of users that already participated to the poll and $(*)$ denotes the present verification session. The equation holds when the user has already participated and the verification process is aborted.

Afterwards, the verifier computes an accumulator A_R such as $A_R = \sigma_2{}^{\mathcal{H}(\mathcal{S}_R)^{-1}}$. The verifier takes uniformly at random $k-1$ integers $\mu_2, \ldots, \mu_k$ and computes l integers $\tau_i \in \mathbb{Z}_p$ for $i \in \{1, \ldots, l\}$, such that $\tau_i = \sum_{j=1}^{k} \mu_j M_{i,j}$

Table 1 List of participating entities in the poll

Poll ID	User presentation token	User one-time proof
	(U_1, Σ_1)	$(T_a{}^{(1)}, T_b{}^{(1)})$
ID_{poll}	(U_2, Σ_2)	$(T_a{}^{(2)}, T_b{}^{(2)})$
	$\vdots$	$\vdots$
	(U_q, Σ_q)	$(T_a{}^{(q)}, T_b{}^{(q)})$

where $M_{i,j}$ is an element of the matrix M. It accepts the presentation token as valid iff Eqs. (3)–(5) lead to the following results:

$$\hat{e}(\sigma_1, g_2) \stackrel{?}{=} X_o \hat{e}(A_R, A)\hat{e}(h_1, C_2') \cdot \mu \tag{3}$$

$$\hat{e}(C_4', g_2) \stackrel{?}{=} \hat{e}(g_1, {C_2'}^{-1}) \tag{4}$$

$$\hat{e}(C_1', g_2) \stackrel{?}{=} \hat{e}(\sigma_2, A^{H(S)^{-1}}) \cdot X_o \cdot \hat{e}(h_1, C_2') \tag{5}$$

where $\mu = \prod_{i=1}^{l} \hat{e}(\gamma_{\rho(i)} {h_1}^{\tau_i}, \omega_i)\hat{e}(\sigma_2, {g_2}^m)$.

- The Trace algorithm is executed by the entity associated to the private key sk_t, hereafter denoted as the tracing authority. The tracing authority decrypts the ciphertext (C_1', C_4'), retrieves $\varpi^* = C_1' \cdot {C_4'}^{\alpha}$, retrieves the value of $({u_j}^*, pk_j, {Y_{u_j}}^{\mathcal{H}(\mathcal{S})^{-1}}, X_{u_j})$ from the issuer table and returns the validity of Eq. (6).

$$\hat{e}(\varpi^*, g_2) \cdot [X_o]^{-1} = \hat{e}({X_u}^{\mathcal{H}(\mathcal{S})^{-1}}, {Y_o}^{sk_t^{-1}}) \tag{6}$$

- Finally, the Judge algorithm, based on ϖ^*, verifies the validity of Eq. (6) to confirm wether the signature Σ was created by user j.

5 Security analysis

We introduce the security model and discuss the resistance of e-$\mathcal{PCS}$ to forgery and anonymity attacks. We also discusss the support of functional properties such as credentials' revocation and multiple issuers' settings.

5.1 Security model

Our security model considers the following two types of adversary:

- *A honest but curious adversary* The adversary is honest, in the sense of generating valid inputs or outputs, during the different steps of the protocol, as well as performing proper computations of the protocol. The adversary is curious, in the sense of gaining extra data from the protocol, such as obtaining credentials and attributes of a given user, or by identifying the requesting user based on the provided presentation tokens. This adversary model can be associated to a service provider, an issuing entity or even a collusion between a curious service provider in collaboration with a curious issuing entity. In all the aforementioned cases, this adversary model affects the validation of the privacy requirements of e-$\mathcal{PCS}$, i.e., with respect to the anonymity and issue-show unlinkability properties.
- *A malicious user* This adversary model assumes a user (or an external entity), trying to override their rights or to attempt a one-time show attacks. In both cases, malicious users are expected to misfollow the associated algorithms (e.g., by providing invalid, falsified or non-authentic inputs).

  A malicious user overriding his rights could refer to an adversary whose attributes do not satisfy the access policy, or he could be a revoked user. We also consider a set of colluding users on the attributes, who do not satisfy the presentation policy and try to merge their attributes to authenticate and access to the SP's resources (i.e., TeSLA cloud domain), in this attack model. The threat model associated to the *malicious user* affects the unforgeability and the one-time show unlinkability requirements of e-$\mathcal{PCS}$.

5.2 Discussion

We prove in this section the correctness of the different e-$\mathcal{PCS}$ algorithms. We also discuss the resistance of the proposed solution with regards to the security and privacy requirements, introduced in Sect. 3.

5.2.1 Correctness

Theorem 1 (Correctness of the e-$\mathcal{PCS}$ construction) *The execution of the* Judge *algorithm, with regard the credential C of user u will always hold true iff, for all* $(params) \leftarrow \mathsf{Setup}(\xi)$, *all pair of public and private keys* $\{(pk_o, sk_o), (pk_u, sk_u)\} \leftarrow \mathsf{KeyGen}(params)$, *all attribute sets* $\mathcal{S}$, *all credentials* $C \leftarrow \mathsf{Issue}(\mathcal{S}, sk_o, pk_u)$, *all claiming predicates* Y *such as* $Y(\mathcal{S}) = 1$ *and all presentation tokens* $\Sigma \leftarrow \mathsf{Show}(C, sk_u, \mathtt{M}, Y)$, ϖ^* *and* $\mathsf{Obtain}(C, sk_u, pk_o, \mathcal{S}) = 1$, $\mathsf{Verify}(\Sigma, m, Y, pk_o)$ *hold true as well.*

Proof Theorem 1 involves the completeness and soundness properties associated to Eqs. (1), (3)–(6). The bilinearity feature of pairing functions guarantees the correctness of Eqs. (1), (4) and (5), (6), such that:

$$\begin{aligned}
\hat{e}(C_1, g_2) &= \hat{e}(x_o \cdot [{X_u}^{s_o \mathcal{H}(\mathcal{S})^{-1}}] \cdot {h_1}^{r}, g_2) \\
&= \hat{e}({g_1}^{s_o}, g_2) \cdot \hat{e}({h_1}^{sk_u s_o \mathcal{H}(\mathcal{S})^{-1}}, g_2) \cdot \hat{e}({h_1}^{r}, g_2) \\
&= \hat{e}(g_1, g_2)^{s_o} \cdot \hat{e}({g_1}^{sk_u \mathcal{H}(\mathcal{S})^{-1}}, {h_2}^{s_o}) \cdot \hat{e}(h_1, {g_2}^{r}) \\
&= X_o \cdot \hat{e}({g_1}^{sk_u \mathcal{H}(\mathcal{S})^{-1}}, Y_o) \cdot \hat{e}(h_1, C_2) \\
\hat{e}(C_4, g_2) &= \hat{e}({g_1}^{-r}, g_2) = \hat{e}(g_1, {g_2}^{-r}) \\
&= \hat{e}(g_1, {C_2}^{-1}) \\
\hat{e}(C_1, g_2) &= \hat{e}(x_o \cdot [{X_u}^{s_o \mathcal{H}(\mathcal{S})^{-1}}] \cdot {h_1}^{r}, g_2) \\
&= \hat{e}({g_1}^{s_0}, g_2) \cdot \hat{e}({g_1}^{\alpha s k_u s_o \mathcal{H}(\mathcal{S})^{-1}}, g_2) \cdot \hat{e}(h_1, {g_2}^{r}) \\
&= X_o \cdot \hat{e}(g_1, {g_2}^{\alpha s k_u s_o \mathcal{H}(\mathcal{S})^{-1}}) \cdot \hat{e}(h1, C_2) \\
&= X_o \cdot \hat{e}({g_1}^{r_m}, ({h_2}^{s_o})^{sk_u / r_m \cdot \mathcal{H}(\mathcal{S})^{-1}}) \cdot \hat{e}(h1, C_2) \\
&= X_o \cdot \hat{e}(\sigma_2, A^{\mathcal{H}(\mathcal{S})^{-1}}) \cdot \hat{e}(h1, C_2)
\end{aligned}$$

The correctness of Eq. (3) is related to the completeness and soundness of the presentation token. The verification of the token will always hold *true* iff $\Sigma = (\Omega, \sigma_1, \sigma_2, C'_1, C'_2, C'_4, A, \mathcal{S}_R)$ is also *true*, with regard to message $\mathtt{M}$ and predicate Y. The correctness of the process relates to the computation of accumulator A_R and the disclosure of the revealed attributes in $\mathcal{S}_R$ in σ_2, such that $A_R = {\sigma_2}^{\mathcal{H}(\mathcal{S}_R)^{-1}}$, and $\mathcal{H}(\mathcal{S}_R) = \prod_{a_i \in \mathcal{S}_R} \mathcal{H}(a_i)^{-1}$. The value of σ_1 can be expressed as follows:

$$\begin{aligned}
\sigma_1 &= C'_1 \cdot B \cdot \mathtt{M}^{r_m} \\
&= C'_1 \cdot \prod_{i=1}^{l} (\gamma'_{\rho(i)})^{v_i} \cdot {g_1}^{r_m m} \\
&= x_o \cdot {X_u}^{s_o \mathcal{H}(\mathcal{S})^{-1}} \cdot {h_1}^{r+r'} \cdot \prod_{i=1}^{l} (\gamma_{\rho(i)})^{(r+r')v_i} \cdot {g_1}^{r_m m}
\end{aligned}$$

We can now validate the correctness of the verification process associated to the presentation token as follows. Let $(r + r')$ by denoted by R. Let the lefthand arguments of Equation 3 by denoted by Ⓢ, such that:

$$\begin{aligned}
Ⓢ &= \hat{e}(x_o \cdot {X_u}^{s_o \mathcal{H}(\mathcal{S})^{-1}} \cdot {h_1}^{r+r'} \cdot \prod_{i=1}^{l} (\gamma_{\rho(i)})^{Rv_i} \cdot \mathtt{M}^{r_m}, g_2) \\
&= \hat{e}(x_o, g_2) \cdot \hat{e}({X_u}^{s_o \mathcal{H}(\mathcal{S})^{-1}}, g_2) \cdot \hat{e}({h_1}^{R}, g_2) \\
&\quad \cdot \hat{e}({g_1}^{r_m m}, g_2) \cdot \hat{e}(\prod_{i=1}^{l} {\gamma_{\rho(i)}}^{Rv_i}, g_2) \\
&= \hat{e}(g_1, g_2)^{s_o} \cdot \hat{e}({g_1}^{sk_u \mathcal{H}(\mathcal{S}_R \cup \mathcal{S}_H)^{-1}}, {g_2}^{\alpha s_o}) \cdot \hat{e}({h_1}^{R}, g_2) \\
&\quad \cdot \hat{e}(\sigma_2, {g_2}^{m}) \cdot \prod_{i=1}^{l} \hat{e}({\gamma_{\rho(i)}}^{Rv_i}, g_2) \\
&= X_o \cdot \hat{e}([{g_1}^{sk_u}]^{\mathcal{H}(\mathcal{S}_R)^{-1} \mathcal{H}(\mathcal{S}_H)^{-1}}, {h_2}^{s_o}) \cdot \hat{e}(h_1, {g_2}^{R}) \\
&\quad \cdot \hat{e}(\sigma_2, {g_2}^{m}) \cdot \prod_{i=1}^{l} \hat{e}(\gamma_{\rho(i)}, {g_2}^{Rv_i}) \\
&= X_o \cdot \hat{e}({g_1}^{\mathcal{H}(\mathcal{S}_R)^{-1}}, [{Y_o}^{sk_u}]^{\mathcal{H}(\mathcal{S}_H)^{-1}}) \cdot \hat{e}(h_1, C'_2) \\
&\quad \cdot \hat{e}(\sigma_2, {g_2}^{m}) \cdot \prod_{i=1}^{l} \hat{e}(\gamma_{\rho(i)}, \omega_i) \\
&= X_o \cdot \hat{e}(A_R, A) \cdot \hat{e}(h_1, C'_2) \cdot \prod_{i=1}^{l} \cdot \hat{e}(\gamma_{\rho(i)} {h_1}^{\tau_i}, \omega_i) \\
&\quad \cdot \hat{e}(\sigma_2, {g_2}^{m})
\end{aligned}$$

Note that the last equality is simplified to $\sum_{i=1}^{l} \tau_i (v_i R) = R \sum_{i=1}^{l} \tau_i v_i = R \cdot 1 = R$, knowing that $\tau_i = \sum_{i=1}^{k} \mu_j M_{i,j}$. Finally, the term $\hat{e}({h_1}^{R}, g_2)$ leads to the following expression: $\hat{e}({h_1}^{R}, g_2) = \prod_{i=1}^{l} \hat{e}({h_1}^{R\tau_i}, {g_2}^{Rv_i})$.

The knowledge associated to sk_t concludes the proof, and allowing decrypting the cyphertext, by validating Equation 6 as follows:

$$\begin{aligned}
\hat{e}(\varpi^*, g_2) \cdot X_o^{-1} &= \hat{e}(C'_1 {C'_4}^{sk_t}, g_2) \cdot {X_o}^{-1} \\
&= \hat{e}([x_o \cdot {X_u}^{S_o \mathcal{H}(\mathcal{S})^{-1}} \cdot {h_1}^{r+r'}] \\
&\quad \cdot {g_1}^{-(r+r')sk_t}, g_2) \cdot {X_o}^{-1} \\
&= \hat{e}(x_o, g_2) \cdot \hat{e}({X_u}^{s_o \mathcal{H}(\mathcal{S})^{-1}} \\
&\quad \cdot {g_1}^{\alpha(r+r')} \cdot {g_1}^{-(r+r')\alpha}, g_2) \cdot {X_o}^{-1} \\
&= \hat{e}({X_u}^{\mathcal{H}(\mathcal{S})^{-1}}, {g_2}^{s_o}) \\
&= \hat{e}({X_u}^{\mathcal{H}(\mathcal{S})^{-1}}, {Y_o}^{sk_t^{-1}})
\end{aligned}$$

□

5.2.2 Unforgeability

The unforgeability property captures the behavior of a non-authorized signing entity. That is, a malicious user

attempts to provide a signed token, that can be correctly verified by the service provider, based on the Verify algorithm. For this purpose, a malicious user could try (1) a credential forgery attack, trying to construct a valid credential, or (2) a presentation token forgery attack, trying to provide a valid presentation token. The adversary can try to gather information from previous issuance and/or presentation sessions. The unforgeability property is formally defined with respect to Theorem 2.

Theorem 2 (Unforgeability) e-$\mathcal{PCS}$ *satisfies the unforgeability property, if for every PPT adversary* $\mathcal{A}$, *there exists a negligible function* ϵ *such that:*

$$Pr[\mathbf{Exp}_{\mathcal{A}}^{unforg}(1^{\xi}) = 1] \leq \epsilon(\xi)$$

where $\mathbf{Exp}_{\mathcal{A}}^{unforg}$ *is the security experiment against the unforgeability property, with respect to Lemma 1, Lemma 2 and Lemma 3 introduced hereafter.*

Proof It relies on proving Lemmas 1, 2, and 3. □

Lemma 1 e-$\mathcal{PCS}$ *is resistant to credential forgery attacks.*

Proof Consider an adversary trying to disrupt the e-$\mathcal{PCS}$ construction by forcing the Obtain algorithm to accept an invalid credential C^* with attributes $\mathcal{S}^*$. It is assumed that the adversary can conduct a polinomially bounded number of queries to the Issue algorithm with different combination of attribute and key pair sets, but ignore the private key of the issuing organization.

Assume the adversary tries to generate a valid credential by perpetrating a forgery attack. To successfully perpetrate the attack, the adversary must confrontate the Computational Diffie Hellman (CDH) assumption. The assumption can be summarized as follows: given a tuple (g, g^a, g^b), were $\{a, b\} \xleftarrow{R} \mathbb{Z}_p$, it is not feasible to compute g^{ab} in polynomial time. We assume now that the adversary considers the credential element C_1. This element is a product of an accumulator over the set of attributes of the user, the private key of the issuing organization x_o and a randomization of the public group element h_1. Knowing this aforementioned randomization is required for deriving the remaining credential elements. Therefore, a successful forgery attack by the adveresary would violate the CDH assumption, hence protecting the e-$\mathcal{PCS}$ construction from credential forgery attacks. □

Lemma 2 e-$\mathcal{PCS}$ *is resistant to presentation tokens forgery attacks.*

Proof For Lemma 2, we consider the following setting. Assume an adversary is allowed to conduct a polynomially bounded number of queries associated to presentation tokens, for any selected signing access policy Y where $Y(\mathcal{S})$ equals one. To successfully perpetrate the attack, the adversary must provide a valid presentation token for a valid credential C accepted by a honest verifier.

To successfully perpetrate the attack, the adversary must confrontate the Computational Diffie Hellman (q-DHE) assumption. The assumption can be summarized as follows. Let $\mathbb{G}$ be a multiplicative cyclic group of a prime order p. Let g be a generator of $\mathbb{G}$. Given a tuple of elements $(g, g_1, \ldots, g_q, g_{q+2}, \ldots, g_{2q})$, such that $g_i = g^{\alpha^i}$, where $i \in \{1, \cdots, q, q+2, \ldots, 2q\}$ and $\alpha \xleftarrow{R} \mathbb{Z}_p$, there is no efficient probabilistic algorithm $\mathcal{A}_{qDHE}$ that can compute the missing group element $g_{q+1} = g^{\alpha^{q+1}}$. The adversary has to violate the aforementioned assumption to provide a valid presentation token. Hence, it is atrighforward that e-$\mathcal{PCS}$ is resistant to credential forgery attacks. □

Lemma 3 e-$\mathcal{PCS}$ *is resistant to collusion attacks.*

Proof Collusion attacks refer to malicious users trying to merge their attributes and certified credentials to provide a valid signature over SP's access policy. We assume that none of the malicious users does possess the whole certified attributes that satisfy the access policy, while their merged attributes permit to satisfy the SP's presentation policy.

To successfully perpetrate the attack, the adversary must be able to generate a valid presentation token for two key pairs (pk_{u_j}, sk_{u_j}) for $j \in \{1, 2\}$, and with respect to a signing predicate such that $Y(\mathcal{S}_j) \neq 1$. The same rationale used in the proof of Lemma 1 applies here. Given the infeasibility of credential forgery attacks against e-$\mathcal{PCS}$, the adversary cannot override the granted rights by conducting a collusion attack using two different credentials. Hence, e-$\mathcal{PCS}$ is resistant to collusion attacks. □

5.2.3 Issue-show unlinkability

The issue-show unlinkability property is formally defined by Theorem 3.

Theorem 3 (Issue-show unlinkability) e-$\mathcal{PCS}$ *satisfies the issue-show unlinkability if no Probabilistic Polynomial Time (PPT) adversary can compute a negligible function* ϵ *such that:*

$$Pr[\mathbf{Exp}_{\mathcal{A}}^{is-unl}(1^{\xi}) = 1] = \frac{1}{2} \pm \epsilon(\xi)$$

where $\mathbf{Exp}_{\mathcal{A}}^{is-unl}$ *is the security experiment against the privacy property, with respect to Lemma 4 introduced hereafter.*

Proof The proof of Theorem 3 is straightforward after the proof of Lemma 4. □

Lemma 4 e-$\mathcal{PCS}$ *is resistant to issue-show unlinkability attacks.*

Proof For Lemma 4, we consider an adversary who is given two keypairs (pk_{u_1}, sk_{u_1}) and (pk_{u_2}, sk_{u_2}), as well as a set of attributes $\mathcal{S}$. We assume that the adversary can conduct a polynomially bounded number of presentation phases, under the role of a verifier, trying to sign either a predicate Y satisfied by $\mathcal{S}$; or a subset of $\mathcal{S}$ for two fixed credentials C_1 associated to $\mathcal{S}$ for pk_{u_1} and C_2 associated to $\mathcal{S}$ for pk_{u_2}. To successfully perpetrate an unlinkability attack, the adversary is required to guess which key pair (pk_{u_j}, sk_{u_j}) w.r.t. a related credential C_j (and for $j \in \{1, 2\}$) used in the presentation procedure, with respect to a fixed signing predicate Y and a set of attributes $\mathcal{S}$. Since a new presentation token for the same message $\mathtt{M}$ and the same access predicate Y is computed from random nonces generated by $\mathcal{C}$, both presentation tokens are identically distributed in both cases, hence confirming that e-$\mathcal{PCS}$ is resistant to issue-show unlinkability attacks. □

5.2.4 One-time show unlinkability

The one-time show unlinkability property is formally defined by Theorem 4.

Theorem 4 (One-time show unlinkability) e-$\mathcal{PCS}$ *satisfies the one-time show unlinkability if no Probabilistic Polynomial Time (PPT) adversary can compute a negligible function* ϵ *such that:*

$$Pr[\mathbf{Exp}_{\mathcal{A}}^{ots-unl}(1^{\xi}) = 1] = \frac{1}{2} \pm \epsilon(\xi)$$

where $\mathbf{Exp}_{\mathcal{A}}^{ots-unl}$ *is the security experiment against the privacy property, with respect to Lemma 5 introduced hereafter.*

Proof The proof of Theorem 4 relies on Lemma 5. □

Lemma 5 e-$\mathcal{PCS}$ *is resistant to one-time show unlinkability attacks.*

Proof The proof of Lemma 5 is mainly based on the correctness of Equation 2. Let us consider a malicious user $\mathcal{A}$ that wants to participate twice to the same poll with the same verifier.

Given two different presentation tokens $\Sigma^{(1)}$ and $\Sigma^{(2)}$ defined respectively as $\Sigma^{(1)} = (\Omega^{(1)}, {\sigma_1}^{(1)}, {\sigma_2}^{(1)}, {C'_1}^{(1)}, {C'_2}^{(1)}, {C'_4}^{(1)}, A^{(1)}, {\mathcal{S}_R}^{(1)})$ and $\Sigma^{(2)} = (\Omega^{(2)}, {\sigma_1}^{(2)}, {\sigma_2}^{(2)}, {C'_1}^{(2)}, {C'_2}^{(2)}, {C'_4}^{(2)}, A^{(2)}, {\mathcal{S}_R}^{(2)})$.

The verifier checks the correctness of Equation 2 as follows:

$$\begin{aligned}
T_a^{(1)} \cdot {T_b}^{(2)} &= \hat{e}({C'_1}^{(1)}, g_2) \cdot \hat{e}(g_1, {C'_2}^{(2)}) \\
&= \hat{e}(x_o \cdot [{X_u}^{s_o \mathcal{H}(\mathcal{S})^{-1}}] \cdot h_1^{(r+r'_1)}, g_2) \cdot \hat{e}(h_1, g_2^{r+r'_2}) \\
&= \hat{e}(x_o \cdot [{X_u}^{s_o \mathcal{H}(\mathcal{S})^{-1}}], g_2) \cdot \hat{e}(h_1^{(r+r'_1)}, g_2) \cdot \hat{e}(h_1^{(r+r'_2)}, g_2) \\
&= \hat{e}(x_o \cdot [{X_u}^{s_o \mathcal{H}(\mathcal{S})^{-1}}] \cdot h_1^{(r+r'_2)}, g_2) \cdot \hat{e}(h_1, g_2^{r+r'_1}) \\
&= \hat{e}({C'_1}^{(2)}, g_2) \cdot \hat{e}(g_1, {C'_2}^{(1)}) \\
&= {T_a}^{(2)} \cdot T_b^{(1)}
\end{aligned}$$

This also proves that the equality in Equation 2 holds with a non negligible probability if both $\Sigma^{(1)}$ and $\Sigma^{(2)}$ values are derived from two different combinations of credentials and the presentation tokens, i.e., if they are linkable. This demonstrates that an adversary cannot participate twice to the same poll, validating the one-time show unlinkability property of e-$\mathcal{PCS}$. □

5.2.5 Anonymity removal

The anonymity removal property refers to the accountability security requirement.

Theorem 5 (Anonymity Removal) e-$\mathcal{PCS}$ *satisfies the traceability property, with respect to Lemma 6 introduced hereafter.*

Proof The proof of Theorem 5 relies Lemma 6. □

Lemma 6 e-$\mathcal{PCS}$ *is resistant to anonymity removal attacks*

Proof Consider an adversary trying to perpetrate an attack that violates the Show ↔ Verify procedure. Assume as well that the tracing authority is unable to trace the identity of the source presentation token. The following two cases can be considered. First, an adversary that forges a valid presentation token. Second, an adversary who knows private key sk_u, i.e., an adversary who is able to derive valid presentation tokens according to Equations 3, 4 and 5, but failing the verification of Equation 6 to avoid the traceability of the attack.

The first case, an adversary that forges a valid presentation token, contradicts the unforgeability property of the e-$\mathcal{PCS}$ construction, already demonstrated in this section. The second case, adversary knowing secret sk_u, is also infeasible,

since Eq. (4) (resp. Eq. 5), C'_4 and C'_2 (resp. C'_1 and C'_2) are verified while being generated with the same randoms. If C'_4, C'_2 and C'_1 along with other signature Σ parameters successfully verify Eq. (3), that means that they all are correlated. Hence, they cannot be falsified while satisfying Eq. (6). Therefore, we can confirm that e-$\mathcal{PCS}$ is resistant against anonymity removal attacks. □

5.3 Functional requirements discussion

Next, we discuss about two main functional properties of e-$\mathcal{PCS}$, namely multiple issuers' settings and credentials' revocation.

5.3.1 Support of multiple issuers

As introduced in Sect. 3, the cloud domain, defined over several TeSLAinstances shared among all institutions, refers to the service provider AC entity. The service provider is responsible for authenticating different users (i.e., TeSLA learners) w.r.t. provided presentation tokens. Recall that a presentation token may be generated while combining several credentials, issued from different issuers, that fulfill the SP's access policy.

For instance, a learner may be enrolled with several institutions, thus holding credentials issued from each different institution. As such, the learner has to be able to derive a valid presentation token, relying on the combination of credentials issued from different issuing entities.

In case users force multiple authorities to issue credentials derived from their attributes, then different sessions will be mapped through the public keys of those users. The unlinkability property of the AC schemes between several issuance sessions extended the credentials' issuance algorithm that to support pseudonym systems and public key masking during the issuance procedure [cf. Kaaniche and Laurent (2016)]. For this purpose, an aggregating algorithm is introduced, to provide the credentials' combination feature. The added algorithm mainly relies on the homomorphism property that permits to merge several credentials' elements (i.e., signed attributes) issued from different issuing entities using different signing keys (i.e., the private keys of different institutions).

As the main issuance phase follows the scheme by Kaaniche et al. (2017), e-$\mathcal{PCS}$ can be extended to support the homomorphism property in order to ensure the public key masking during the issuance phase. This ensures the support of multiple issuers' settings, mainly inherited from the construction in Kaaniche and Laurent (2016).

5.3.2 Credentials revocation

Credentials' revocation is a main issue in AC mechanisms. As detailed in Sect. 2, to effectively generate and accurately verify presentation tokens, the most recent revocation information has to be gathered from the revocation authority, by the user, respectively the verifier. In other words, the revocation authority will be in charge of conducting the revocation of credentials previously issued to a given user. The revocation authority is also in charge of maintaining the database containing valid credentials, and disseminating this information to the remainder entities. In case of revocation, the revoked credential will not be allowed to derive valid presentation tokens anymore.

In order to support periodical users' revocation, e.g., induced by the necessity of accepting systems' applications deadlines, e-$\mathcal{PCS}$ considers the introduction of a validity attribute. The validity attribute has to be signed by the issuing entity and involved in the user's credential. This information is later requested by the verifier and has to be included in the presentation token provided by the user.

6 Conclusion

We have detailed an AC scheme for e-assessment services called e-$\mathcal{PCS}$. ACs are cryptographic mechanisms that allow users to obtain certified credentials for their attributes from trusted issuers. Later, users can derive presentation tokens that reveal only the required information, while satisfying service providers' access policies.

The scheme presented in this paper guarantees the use of one-time show credentials, i.e., credentials that can be shown only once. The goal is to avoid that the originating user ends being traced from one transaction to another (i.e., as happens in multi-show credential schemes, in which users can use their credentials multiple times, hence allowing tracking and profiling of users). Under this context, e-$\mathcal{PCS}$ considers e-assessment opinion polls scenarios, in which one-time show properties are relevant to avoid tracking of e-assessment learners. To mitigate cheating, the scheme is provided with two extra procedures: attribute revocation and anonymity removal. A detailed security analysis has been conducted, to prove the correctness of our scheme, as well as some other properties, such as unforgeability, privacy and anonymity removal.

The features of e-$\mathcal{PCS}$ allow e-assessment systems to perform privacy-friendly access control, in order to certify that users are allowed to access a resource because they own some attributes required by the verifier, without revealing their identity. Given traditional assessment principles, where learners receive personalized grades through identity validation, anonymous certification can be applied as

a complement of hosting course material of the e-assessment system not needing the true learners identity to decide whether they should have access to the course material or not. The system can require non-intrusive information, such as whether the learner is enrolled at the university giving the course, or whether the learner has registered for the course. These two items correspond to the two attributes that would be checked in the context of AC. By doing so, we exclude the technical possibility to track the learners' activity, as well as to exclude the profiling of learners according to their hours of activity, the frequency of their access to the course material, etc., hence significantly improving the learners' privacy.

Another way of integrating e-$\mathcal{PCS}$ into e-assessment systems is its addition in the broker of privacy filters as a post processing filtering tool for completed e-assignments. After a learner takes an e-assessment, the assignment is sent to various external e-assessment instruments associated to the list of privacy filters, including some anti-cheating post-processing tools (e.g., to check whether an assignment contains plagiarism). If the assignment is sent along with the identifier (or the pseudo-identifier) of the learner, it becomes technically possible for the system to keep track of a learner's data over time. In the case where e-$\mathcal{PCS}$ is used instead, relying on the aforementioned attributes, it becomes impossible for instruments to perform such tracking and correlation, hence enhancing the learners' privacy.

Acknowledgements This work is supported by the H2020-ICT-2015/H2020-ICT-2015 TeSLA project An Adaptive Trust-based e-assessment System for Learning, number 688520. Authors acknowledge as well support from the European Commission (H2020 SPARTA project), under Grant agreement 830892.

References

Aïmeur E, Hage H (2010) Preserving learners privacy. In: Advances in intelligent tutoring systems. Springer, pp 465–483

Aïmeur E, Hage H, Onana FSM (2008) Anonymous credentials for privacy-preserving e-learning. In: E-Technologies, 2008 international MCETECH conference on, IEEE, pp 70–80

Beimel A (1996) Secret sharing and key distribution. In: Research thesis

Belguith S, Kaaniche N, Laurent M, Jemai A, Attia R (2017) Constant-size threshold attribute based signcryption for cloud applications. In: 14th international conference on security and cryptography (SECRYPT 2017), vol. 6, pp 212–225

Belguith S, Kaaniche N, Laurent M, Jemai A, Attia R (2018a) Phoabe: securely outsourcing multi-authority attribute based encryption with policy hidden for cloud assisted IOT. Comput Netw 133:141–156

Belguith S, Kaaniche N, Russello G (2018b) Pu-abe: Lightweight attribute-based encryption supporting access policy update for cloud assisted iot. In: 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), pages 924–927. IEEE

Brands SA (2000) Rethinking public key infrastructures and digital certificates: building in privacy. MIT Press, Cambridge

Camenisch J, Lysyanskaya A (2001) An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Advances in cryptology EUROCRYPT 2001. Springer, pp 93–118

Camenisch J, Mödersheim S, Sommer D (2010) A formal model of identity mixer. Form Methods Ind Crit Syst 198–214

Chaum D (1985) Security without identification: transaction systems to make big brother obsolete. Commun ACM 28(10):1030–1044

Cole J, Foster H (2007) Using Moodle: teaching with the popular open source course management system. O'Reilly Media, Inc, Newton

Gathuri JW, Luvanda A, Matende S, Kamundi S (2014) Impersonation challenges associated with e-assessment of university students. J Inf Eng Appl 4(7):60–68

Herranz J, Laguillaumie F, Libert B, Rafols C (2012) Short attribute-based signatures for threshold predicates. In: Topics in cryptology—CT-RSA 2012

IBM (2018) IBM identity mixer. https://www.zurich.ibm.com/identity_mixer/

Kaaniche N, Laurent M (2016) Attribute-based signatures for supporting anonymous certification. In: European symposium on research in computer security. Springer, pp 279–300

Kaaniche N, Laurent M, Rocher P-O, Kiennert C, Garcia-Alfaro J (2017) PCS, a privacy-preserving certification scheme. In: Data privacy management, and security assurance—12th international workshop, DPM 2017, Oslo, Norway, September, 2017, Lecture notes in computer science. Springer

Karchmer M, Wigderson A (1993) On span programs. In: In Proc. of the 8th IEEE Structure in Complexity Theory

Kiennert C, Kaaniche N, Laurent M, Rocher P-O, Garcia-Alfaro J (2017a) Anonymous certification for an e-assessment framework. In: Nordic Conference on Secure IT Systems, pages 70–85. Springer

Kiennert C, Rocher PO, Ivanova M, Rozeva A, Durcheva M, Garcia-Alfaro J (2017b) Security challenges in e-assessment and technical solutions. In: 8th international workshop on interactive environments and emerging technologies for eLearning, 21st international conference on information visualization, London, UK

Kim S-K, Huh J-H (2018) A study on the LMS platform performance and performance improvement of k-moocs platform from learner's perspective. In: Advanced multimedia and ubiquitous engineering. Springer, pp 781–786

Li J, Au MH, Susilo W, Xie D, Ren K (2010) Attribute-based signature and its applications. ASIACCS '10

Lindell Y, Katz J (2014) Introduction to modern cryptography. Chapman and Hall/CRC, New York

Liu X, Xia Y, Sun Z (2017) Provably secure attribute based signcryption with delegated computation and efficient key updating. KSII Trans Internet Inf Syst 11(5):2646

Maji HK, Prabhakaran M, Rosulek M (2011) Attribute-based signatures. In: Cryptographers track at the RSA conference. Springer, pp 376–392

Okamoto T, Takashima K (2011) Efficient attribute-based signatures for non-monotone predicates in the standard model. PKC'11

Paquin C, Zaverucha G (2011) U-prove cryptographic specification v1. 1. Technical report, Microsoft Corporation

Rescorla E, Dierks T (2008) The transport layer security (TLS) protocol version 1.2. RFC 5246

Shahandashti S, Safavi-Naini R (2009) Threshold attribute-based signatures and their application to anonymous credential systems. AFRICACRYPT '09

TeSLA Consortium (2016) Trust based authentication & authorship e-assessment analysis. http://tesla-project.eu/

Wu X, Wu J (2019) Criteria evaluation and selection in non-native language MBA students admission based on machine learning methods. J Ambient Intell Human Comput

Xu Q, Tan C, Fan Z, Zhu W, Xiao Y, Cheng F (2018) Secure data access control for fog computing based on multi-authority attribute-based signcryption with computation outsourcing and attribute revocation. Sensors 18(5):1609

Zhang Y, Feng D (2012) Efficient attribute proofs in anonymous credential using attribute-based cryptography. In: Proceedings of the 14th international conference on information and communications security, ICICS'12