

Dominic Deuber, Matteo Maffei, Giulio Malavolta\*, Max Rabkin, Dominique Schröder, and Mark Simkin

# Functional Credentials

**Abstract:** A functional credential allows a user to anonymously prove possession of a set of attributes that fulfills a certain policy. The policies are arbitrary polynomially computable predicates that are evaluated over arbitrary attributes. The key feature of this primitive is the delegation of verification to third parties, called designated verifiers. The delegation protects the *privacy of the policy*: A designated verifier can verify that a user satisfies a certain policy without learning anything about the policy itself. We illustrate the usefulness of this property in different applications, including outsourced databases with access control. We present a new framework to construct functional credentials that does not require (non-interactive) zero-knowledge proofs. This is important in settings where the statements are complex and thus the resulting zero-knowledge proofs are not efficient. Our construction is based on any predicate encryption scheme and the security relies on standard assumptions. A complexity analysis and an experimental evaluation confirm the practicality of our approach.

**Keywords:** Anonymous Credentials, Anonymous Authentication

DOI 10.1515/popets-2018-0013

Received 2017-08-31; revised 2017-12-15; accepted 2017-12-16.

## 1 Introduction

Anonymous credentials were proposed by Chaum [25] and first fully realized in the seminal work by Camenisch and Lysyanskaya [17]. This primitive allows users to prove possession of a set of attributes that fulfills a certain policy without disclosing anything about the at-

tributes beyond what is disclosed by the policy itself. In this work, we suggest *functional* credentials (FC) as a generalization and unification of (anonymous) credentials and their derivatives. In a functional credential, policies are arbitrary polynomially computable functions that are evaluated on arbitrary attributes. Functional credentials allow the *delegation of verification* to third parties, called designated verifiers. This property states that a designated verifier can verify that a certain user holds a credential encoding attributes that satisfy the desired policy but *does not learn anything about the policy* itself. The same holds also for the carrier of the credential. Beyond constituting a theoretically interesting primitive in its own right, functional credentials are an ideal primitive in the context of privacy-enhancing technologies in cloud-based scenarios where the service provider (the designated verifier) is in charge of performing access control. In the following, we exemplify some applications and discuss how our construction improves the efficiency of known solutions.

VERIFIABLE DATABASES WITH ACCESS CONTROL. Recently, several works have tackled the problem of *oblivious* outsourced storage for data-sharing applications with access control being enforced by an honest-but-curious cloud server, such as, among others, group oblivious RAM as suggested by Maffei et al. [37]. Obliviousness in this setting means the cloud provider does not even learn the access pattern to the data. Policy hiding is crucial in this setting, since obliviousness would be broken if the cloud provider would learn the access control policy of the retrieved entry. The construction of Maffei et al. [37] uses Groth-Sahai zero-knowledge proofs [32] to enforce the access control. Since the ciphertexts are very large, these proofs are quite expensive. Replacing Groth-Sahai proofs with functional credentials improves the efficiency of the scheme by at least one order of magnitude. A comprehensive discussion and comparison is given in Section 4.

OFFLINE CREDIT SCORE SYSTEMS. Before releasing a credit card or opening an account, banks want to check that the client's financial status is above a certain threshold. This is achieved by contacting a credit bureau, which tracks all relevant user activities and, based on them, classifies the client's financial status. Interestingly, the formula used by the credit bureau is

---

**Dominic Deuber:** Friedrich-Alexander-Universität Erlangen-Nürnberg, E-mail: dd@cs.fau.de

**Matteo Maffei:** TU Wien, matteo.maffei@tuwien.ac.at

**\*Corresponding Author: Giulio Malavolta:** Friedrich-Alexander-Universität Erlangen-Nürnberg, E-mail: giulio.malavolta@fau.de

**Max Rabkin:** E-mail: max.rabkin@gmail.com

**Dominique Schröder:** Friedrich-Alexander-Universität Erlangen-Nürnberg, E-mail: dominique.schroeder@fau.de

**Mark Simkin:** Aarhus University, E-mail: simkin@cs.au.dk

kept confidential. Functional credentials offer a privacy-preserving solution to this problem: The credit bureau plays the role of the issuer and initially provides the bank, which plays the role of the verifier, with a delegation token encoding the confidential credit policy, and the client with a credential encoding her financial status. The client can later prove to the bank that her credit score satisfies the bank’s requirements, without any further interaction with the credit bureau.

DATING SYSTEMS. Here a central authority certifies the attributes of each user, providing her with the respective credential. A user can then upload a delegation token encoding her partner preferences to the online dating system, the designated verifier. Other users, playing the role of the prover, can determine whether their attributes match the user preferences, which preferably should be concealed from the service provider as well as from the other users.

## 1.1 Our Techniques

The construction of an efficient functional credential scheme is non-trivial, due to the large expressiveness and flexibility of the supported policies. Ideally, we would like to avoid any restriction on the predicates that can be encoded as verification policies while at the same time guaranteeing the privacy of the policy. Such expressive systems often require very heavy theoretical tools, for example, a straightforward instantiation of functional credentials might build on top of zero-knowledge proofs showing the validity of the encrypted policy for the possessed credentials. However, this would require the computation of zero-knowledge proofs for operations over encrypted data (i.e., on fully homomorphic encryption schemes [30]), which would lead to prohibitively expensive proofs. In order to avoid costly zero-knowledge proofs, we present a scheme almost exclusively built on top of predicate encryption. We leverage the fact that predicate encryption provides a very natural way to prove statements: Given a ciphertext encoding a given policy, a prover can simply decrypt such a ciphertext to convince a verifier that he knows a key for a set of attributes that matches the policy. Our construction is largely inspired by the work of Parno, Raykova and Vaikuntanathan [41], who build a verifiable computation scheme from any attribute-based encryption, basing on a similar observation. Since almost all anonymous credentials are based on encrypted or committed signatures in combination with zero-knowledge proofs,

we view our approach as a substantial paradigm shift in the context of anonymous credentials. In the following, we discuss a strawman approach and explain the main ideas behind our solution.

A STRAWMAN APPROACH BASED ON PREDICATE ENCRYPTION. A predicate encryption scheme allows one to embed an arbitrary set of attributes  $\mathcal{I}$  in the ciphertext and distribute decryption keys for predicates  $f$ . The message is disclosed to users holding a key for a certain predicate  $f$  only if  $f(\mathcal{I}) = 1$ . Additionally, the primitive guarantees that the set of attributes  $\mathcal{I}$  is kept private, except for the information trivially revealed by the validity of the predicate for the encoded attributes. This allows us to encode access control policies as classes of attributes and users’ permissions as predicates over these classes to obtain the desired expressiveness. The straightforward approach to verify credentials is to prove in zero-knowledge the successful decryption of a given ciphertext (for instance by means of Groth-Sahai proofs [32]). However, the proof would scale with the size of the ciphertext of a predicate encryption making it unusable for practical purposes.

INTERACTIVE PROOF OF DECRYPTION. Improving over the efficiency of the verification protocol requires eliminating the dependency between the generic zero-knowledge frameworks and the policy encoded in the ciphertext. To this extent we sacrifice the non-interactivity of the zero-knowledge proof to obtain a 3-round interactive protocol for showing the credential in an anonymous fashion. Leveraging interaction is natural in cloud-based services, given the online nature of service providers. Our protocol closely follows the authentication proof by Brandt et al. [10]: To prove the knowledge of a key, the client simply decrypts the encryption of a random message (sent by the verifier) and sends a commitment of the resulting plaintext to the verifier. The verifier reveals afterwards the randomness that he used to compute the ciphertext so that the client can locally verify that it was well formed. If this holds true, the client sends the opening information. The verifier is now convinced of the possession of the key if the opening reconstructs to the original plaintext of the ciphertext that he sent to the client. This simple technique removes the occurrence of generic zero-knowledge proofs over predicate encryption schemes.

DELEGATION OF VERIFICATION. We realize our delegated verification protocol with a technique very similar to the one described above. Intuitively, a verification token is an encryption of a fixed message on the encoding

of the desired policy, together with the respective signature of the issuer. While the signature vouches for the authenticity of the token, the ciphertext allows the verifier to run a slightly divergent verification algorithm. Exploiting the homomorphic properties of the predicate encryption scheme, the designated verifier can modify the plaintext and rerandomize the ciphertext to obtain a correctly distributed instance and execute the standard verification algorithm. Beyond the verification of a signature, the two interactive protocols are essentially equivalent in terms of communication and computation.

**INSTANTIATION.** A nice feature of our scheme is that it makes only *black-box* use of the underlying predicate encryption scheme. While the original motivation for this was to avoid expensive zero-knowledge proofs, being independent from the implementation of the primitive gives us the possibility to instantiate our protocol with *any* predicate encryption scheme. In particular, this means that our construction can directly benefit from future advancements in the field. Currently known predicate encryption schemes follow the blueprint of Katz, Sahai, and Waters [34] and they only support predicates of the family of inner products. Although the authors show a generic transformation from inner products to boolean formulae, this imposes some limitations on the resulting family of predicates: One cannot efficiently encode negations of non-boolean attributes and the size of the formula grows exponentially with the maximum number of literals in each disjunctive clause. Therefore the expressiveness of the scheme is limited to boolean formulae in their  $k$ -CNF, for some fixed  $k$ . The implementation of our construction suffers from the same shortcomings. For the reason specified above, in this paper we present our protocol in its full generality and we show some concrete examples of policies that we can encode given the current knowledge of predicate encryption schemes.

## 1.2 Our Contribution

The contributions of this work can be summarized as follows:

- We put forward the notion of functional credentials and formalize the corresponding security notions in terms of cryptographic games. Functional credentials subsume all known credentials, such as anonymous, delegatable, or attribute-based credentials and we view our formalization as a natural unification, along the lines of functional signatures [3, 8].

- We propose a construction that is secure in the standard model based on black-box cryptographic primitives. Although generic, our scheme enjoys a very efficient instantiation in the random oracle model.
- We demonstrate the feasibility of our approach with a C++ implementation. Even for large parameters, the computation and verification of a credential takes only a couple of seconds.
- We show the effectiveness of our techniques on a recently proposed oblivious outsourced storage scheme [37]: We instantiate the authentication mechanism with functional credentials, obtaining better scalability and an improvement in efficiency of at least one order of magnitude.

## 1.3 Related Work

Anonymous credentials were originally envisioned by Chaum [24] and first fully, and efficiently, realized by Camenisch and Lysyanskaya [17]. Such credentials are an enabling technology for anonymous attributed-based access control in internet services. They allow users to prove possession of a set of attributes fulfilling certain policies in a privacy-preserving way without disclosing anything more about the attributes than what is already disclosed by the policy itself. These credentials neither reveal any unnecessary information about the user, nor are separate authentications by the same user linkable by the verifier. Anonymous credentials are one of the few advanced cryptographic primitives that has made its way into practice, e.g., in the form of IBM’s Identity Mixer [20].

Following the seminal paper of Camenisch and Lysyanskaya, subsequent work has shown how to construct anonymous credentials using  $\Sigma$ -protocols [19], Groth-Sahai proofs [32], or other approaches [5]. Apart from being interesting in their own right, they enjoy a wide range of features, such as delegatability [1], blacklistability [2], and revocability [14]. The concept of anonymous credentials was also extended to support hierarchical delegation of credentials [4] and decentralized credential systems, where the assumption of a trusted credential issuer was relaxed in the work of Camenisch and Lehmann [16].

Recently, Hanser and Slamanig [33] (then revised by Fuchsbaauer et al. [29]) proposed the first credential system where the communication complexity for credential showing is independent of the number of its attributes. Unfortunately, their scheme only supports conjunctive statements and thus lacks expressiveness.

Other works [12, 15] obtained similar results in stronger simulation-based models.

The idea of a hidden policy was introduced in the context of hidden credentials [9], where the anonymity of the user is guaranteed by the underlying anonymous identity-based encryption scheme. In the work of Bradshaw et al. [9] and follow up works [28] complex policies are handled by encoding disjunctions and conjunctions in a secret sharing of the encrypted message. Even though their approach is very efficient, it has the fundamental drawback of not being collusion resistant. That is, two users can combine their attributes to decrypt ciphertexts that neither could decrypt on its own. In the work of Camenish et al. [13] the notion of oblivious transfer with hidden access policies is introduced. Their approach can be seen as a form of implicit authentication, where a user has access to some data if he has credentials that fulfill a policy attached to a ciphertext. There is no verifier that is explicitly notified about the success of this authentication. In this implicit authentication the scheme is secure against malicious issuers, but turning their scheme into an explicit authentication scheme loses security against malicious issuers as we will show in this work (see Section 2.3). Although their approach extended to the explicit authentication setting comes close to our desired notion of functional credentials, it still lacks expressiveness, since the policies are limited to eligible subsets of binary attributes. Camenisch et al. [11] proposed a scheme with improved expressiveness, but its realization is specific to the predicate encryption scheme of Nishide et al. [38], which hides the policy embedded in a ciphertext only partially. Both of the aforementioned schemes rely on  $q$ -type assumptions, i.e., parametric in the number of queries of the adversary, making it hard to analyze the concrete security of the schemes.

The problem of delegated verification has been first proposed, but not thoroughly formalized, in the work of Wei and Ye [44]. Their solution leverages the homomorphic properties of the predicate encryption scheme of Katz et al. [34], but only achieves semi-honest security. In a recent work Kolesnikov et al. [35] combine attribute-based credentials and attributed-based encryption to achieve a new notion of attribute-based key exchange with the goal of letting a server establish a shared secret key with a client if he has a set of certified attributes satisfying the server’s policy. The authors explicitly state policy hiding as an open problem. To the best of our knowledge, none of the currently known constructions provides a general-purpose, fully expressive,

fully secure attribute-based credential system that supports delegated verification and policy hiding.

**COMPARISON.** We outline an asymptotic comparison of our generic construction with the most prominent credential schemes in the literature in Table 1. We provide the reader with an overview over the costs associated with each scheme, in terms of computation and communication, and over the expressiveness of each scheme. Our construction is the first to fully achieve the notion of privacy of the policy without restricting the domain of access control policies to selective attribute disclosure. Here the caveat is that our approach relies generically on predicate encryption and current instantiations support only inner-product predicates, i.e., the class of predicates that the scheme can encode is restricted to inner products over a finite field. This means that, in order to obtain full expressiveness, one must apply the transformation described by Katz et al. [34]. Such a transformation introduces an exponential factor in the number of literals of each disjunctive clause of the resulting formula. As an example, one can encode boolean formulae in their 3-CNF with a cubic blowup. Nevertheless, we believe that our construction represents an interesting theoretical advancement in the landscape of anonymous credentials, due to its simplicity and generality.

## 2 Functional Credentials

In this section, we introduce the notation and the definition of functional credentials and the corresponding notion of security. We denote by  $\lambda \in \mathbb{N}$  the security parameter and we address any function that is *negligible* in the security parameter with  $\text{negl}(\lambda)$  and any polynomial function with  $\text{poly}(\lambda)$ . We say that an algorithm is PPT if it is modelled as a probabilistic Turing machine whose running time is bounded by some polynomially bounded function in  $\lambda$ . We denote by  $\mathcal{A}(\cdot; r)$  the execution of the machine  $\mathcal{A}(\cdot)$  with a fixed random tape  $r$ . Given a set  $S$ , we denote by  $x \leftarrow S$  the sampling of an element uniformly at random from  $S$ . We denote an interactive protocol between algorithms  $A$  and  $B$  as  $\langle A, B \rangle$ . The set  $\{1, \dots, n\}$  is abbreviated as  $[n]$ .

### 2.1 Definition of Functional Credentials

A basic credential system is composed of *users*, *verifiers*, and *issuers*. The latter can issue credentials to users relative to some attributes. Users can prove to verifiers that

Scheme	Assumptions	Parameters ( $\alpha$ attr.)		Issuing		Showing ( $t$ of $\alpha$ )		Policy	Properties		
		CRS	Credential	Issuer	User	Verifier	User		Anon.	Unforg.	PH
[18]	sRSA	$O(\alpha)$	$O(1)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	R	✓	✓	—
[19]	LRSW	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	R	✓	✓	—
[21]	$q$ -ADHSDH	$O(1)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(1)$	S	✓	✓	—
[22]	XDH	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(t)$	$O(t)$	S	✓	✓	—
[12]	SXDH	$O(\alpha)$	$O(1)$	$O(\alpha)$	$O(\alpha)$	$O(t)$	$O(\alpha - t)$	S	(✓)	✓	—
[29]	SXDH	—	$O(1)$	$O(\alpha)$	$O(\alpha)$	$O(t)$	$O(\alpha - t)$	S	(✓)	✓	—
[9]	BDH	ROM	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(t)$	—	R	✓	—	Part.
[13]	BDH,XDH,SDH	$O(\alpha)$	$O(1)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	S	(✓)	✓	✓
[11]	SXDH,SFP	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(\alpha)$	$O(1)$	S	(✓)	✓	Part.
[45]	Gen.	$O(\log^k \alpha)$	$O(\log^k \alpha)$	$O(\log^k \alpha)$	$O(\log^k \alpha)$	$O(\log^k \alpha)$	$O(\log^k \alpha)$	k-CNF	✓	Sel.	—
[44]	GGM	$O(\log^k \alpha)$	$O(\log^k \alpha)$	$O(\log^k \alpha)$	$O(\log^k \alpha)$	$O(\log^k \alpha)$	$O(\log^k \alpha)$	k-CNF	HbC	HbC	HbC
[35]	Gen.	$O(\log^c \alpha)$	$O(\log^c \alpha)$	$O(\log^c \alpha)$	$O(\log^c \alpha)$	$O(\log^c \alpha)$	$O(\log^c \alpha)$	R	✓	✓	—
Sec. 3	Gen.	$O(\log^k \alpha)$	$O(1)$	$O(\log^k \alpha)$	$O(\log^k \alpha)$	$O(\log^k \alpha)$	$O(\log^k \alpha)$	k-CNF	✓	✓	✓

**Table 1.** Asymptotic comparison of the most popular credential schemes. We show the assumptions that the various schemes are proven against: The strong RSA (sRSA), the Lysyanskaya, Rivest, Sahai, and Wolf (LRSW), the  $q$ -Asymmetric Double Hidden Strong Diffie-Hellman ( $q$ -ADHSDH), the external Diffie-Hellman (XDH), the strong external Diffie-Hellman (SXDH), the bilinear Diffie-Hellman (BDH), and the simultaneous flexible pairing (SFP). Here Gen. means generic assumption, GGM stands for Generic Group Model, and the presence of a random oracle is denoted by ROM. The policy parameter measures the expressiveness of the policy that the scheme support: R stands for any arbitrary polynomially computable relation over attributes, whereas S denotes the selective disclosure of a subset of the total attributes. k-CNF denotes formulae in their CNF with at most  $k$  literals in each clause. We denote by  $\log^c(\alpha)$  a polylogarithmic factor in  $\alpha$ , for some constant  $c$  that depends on the underlying encryption scheme used. Whenever we write  $\log^k(\alpha)$  we refer to  $k$  as the bound on the number of literals of the formula. We also compare the schemes in terms of the properties that they achieve: Anonymity, Unforgeability, and Policy-Hiding. Here (✓) means that the anonymity is guaranteed also in the presence of a malicious issuer, Part. means that the scheme partially achieves the corresponding property, Sel. means that it achieves the selective version of it, and HbC denotes the Honest-but-Curious settings.

their credentials satisfy a certain policy. Verification can be delegated to third parties, called *designated verifiers*, who can verify that a certain user satisfies a certain policy but cannot learn anything about this policy. The sets of users and verifiers (including designated verifiers) are potentially dynamic and can grow over time. In our model, the issuer assigns to each user a set of attributes  $A$  and the verifier can later on check that  $A$  satisfies a certain policy  $f$ . We denote the attribute universe by  $\Theta$  and the family of circuits that can be computed over  $\Theta$  by  $\Phi$ . Note that we allow only the issuer to produce valid delegation tokens to avoid trivial attacks on the privacy of the users; see Section 2.3 for a more extensive discussion on the matter.

**Definition 1** (Functional Credential). *A functional credential scheme  $FC = (\text{CKGen}, \text{GrantCred}, \text{Del}, \langle \text{ShowCred}, \text{VrfyCred} \rangle, \langle \text{DelShowCred}, \text{DelVrfyCred} \rangle)$  for an attribute universe  $\Theta$  and a family of policies  $\Phi$  consists of the following PPT algorithms and protocols:*

$(\text{osk}, \text{opk}) \leftarrow \text{CKGen}(1^\lambda)$ : *The key generation algorithm gets as input the security parameter  $1^\lambda$  and it outputs a key pair  $(\text{osk}, \text{opk})$  for an issuer.*

$\text{cred} \leftarrow \text{GrantCred}(\text{osk}, A)$ : *The grant credential algorithm gets input the secret key  $\text{osk}$  of an issuer and a*

*non-empty set of attributes  $A \subseteq \Theta$  and it outputs a credential  $\text{cred}$  for the corresponding set of attributes.*

$\text{tok} \leftarrow \text{Del}(\text{osk}, f)$ : *The delegation algorithm gets as input the private key  $\text{osk}$  of an issuer and a policy  $f \in \mathcal{C}$ . The algorithm outputs a verification token  $\text{tok}$ .*

$b \leftarrow \langle \text{ShowCred}(\text{opk}, \text{cred}, f), \text{VrfyCred}(\text{opk}, f) \rangle$ : *This interactive protocol is run by a user and a verifier. ShowCred takes as input the public key  $\text{opk}$  of an issuer, a credential  $\text{cred}$ , and a policy  $f$ , while VrfyCred takes as input the public key  $\text{opk}$  of an issuer and a policy  $f$ . At the end of the execution, VrfyCred outputs either 0 or 1.*

$b \leftarrow \langle \text{DelShowCred}(\text{opk}, \text{cred}, \text{tok}), \text{DelVrfyCred}(\text{opk}, \text{tok}) \rangle$ : *This interactive protocol is run by a user and a designated verifier. DelShowCred takes as input the public key  $\text{opk}$  of an issuer, a verification token  $\text{tok}$ , and a credential  $\text{cred}$ , while DelVrfyCred takes as input the public key  $\text{opk}$  of an issuer and a verification token  $\text{tok}$ . At the end of the execution, DelVrfyCred outputs either 0 or 1.*

The definition of correctness for functional credentials looks as follows.

**Definition 2** (Correctness). *A functional credential scheme  $FC$  is correct if for all  $\lambda \in \mathbb{N}$ , for all  $(\text{osk}, \text{opk}) \in \text{CKGen}(1^\lambda)$  for all  $A \subseteq \Theta$ , for all  $\text{cred} \in$*

$\text{GrantCred}(\text{osk}, A)$ , for all  $f \in \mathcal{C}$  such that  $f(A) = 1$ , and for all  $\text{tok} \in \text{Del}(\text{osk}, f)$  it holds that

$$\Pr[1 \leftarrow \langle \text{ShowCred}(\text{opk}, \text{cred}, f), \text{VrfyCred}(\text{opk}, f) \rangle] = 1$$

and

$$\Pr \left[ 1 \leftarrow \left\langle \begin{array}{l} \text{DelShowCred}(\text{opk}, \text{cred}, \text{tok}), \\ \text{DelVrfyCred}(\text{opk}, \text{tok}) \end{array} \right\rangle \right] = 1.$$

## 2.2 Security of Functional Credentials

In this section, we propose our security model for functional credentials and formalize the definitions according to the game-based paradigm. We formalize the information that the adversary can gather from a running system as oracles that the adversary can query adaptively and at any time. Furthermore, the adversary is allowed to open interleaving sessions with each oracle. Note that we consider a single issuer but the definitions extend easily to multiple issuers.

**GLOBAL VARIABLES AND ORACLES.** In the following security games, the adversary has access to a subset of the following oracles:  $\mathcal{O}^{\text{User}}$  adds honest users to the system,  $\mathcal{O}_{\text{osk}}^{\text{GrantCred}}$  assigns a credential encoding a given a set of attributes to an input user,  $\mathcal{O}^{\text{Corrupt}}$  allows the adversary to corrupt any user. Furthermore, the adversary can see arbitrarily many delegation tokens via the oracle  $\mathcal{O}_{\text{osk}}^{\text{Del}}$  and is also allowed to play the role of the verifier in both the verification and delegated verification protocol through the oracles  $\mathcal{O}^{\text{ShowCred}}$  and  $\mathcal{O}^{\text{DelShowCred}}$ . The oracles  $\mathcal{O}_b^{\text{Anon}}$ ,  $\mathcal{O}_b^{\text{AnonDel}}$ , and  $\mathcal{O}_b^{\text{PH}}$  are used later in the security experiments to compute the challenges for the adversary. We assume that two global lists of users are shared among the oracles:  $\mathcal{H}$  is a list of honest users and  $\mathcal{C}$  is a list of corrupted users. Additionally, we keep a list of user-credential pairs  $\mathcal{Q}$  and a list of policy-token pairs  $\mathcal{L}$ . We assume  $\mathcal{Q}$  and  $\mathcal{L}$  to be available to all of the oracles. We denote the  $i$ -th entry of  $\mathcal{Q}$  by  $\mathcal{Q}[i]$ . For ease of notation, we implicitly assume that every oracle takes as input the public key  $\text{opk}$ . The oracles are formalized in the following.

$\mathcal{O}^{\text{User}}(\text{id})$ : The user adding oracle takes as input a user identity  $\text{id}$ . If  $\text{id} \in \mathcal{H}$  or  $\text{id} \in \mathcal{C}$  it returns  $\perp$ , else it creates a fresh entry  $\text{id}$  in the list of honest users  $\mathcal{H}$ .

$\mathcal{O}^{\text{Corrupt}}(\text{id})$ : The input of the corruption oracle is a user identity  $\text{id}$ . If  $\text{id} \notin \mathcal{H}$  it returns  $\perp$ , else it moves the entry corresponding to  $\text{id}$  from the list of honest users  $\mathcal{H}$  to the list of corrupted users  $\mathcal{C}$ . Then, for all items of the form  $(\text{id}, A_i, \text{cred}_i) \in \mathcal{Q}$ , the oracle returns  $\text{cred}_i$ .

$\mathcal{O}_{\text{osk}}^{\text{Del}}(f)$ : On input a policy  $f$  the delegation oracle returns  $\text{tok} \leftarrow \text{Del}(\text{osk}, f)$  and adds  $(f, \text{tok})$  to the list of tokens  $\mathcal{L}$ .

$\mathcal{O}_{\text{osk}}^{\text{GrantCred}}(\text{id}, A)$ : The credential granting oracle takes as input a user identity  $\text{id}$  and a non-empty set of attributes  $A \subseteq \Theta$ . If  $\text{id} \notin \mathcal{H}$  and  $\text{id} \notin \mathcal{C}$  it returns  $\perp$ , else it runs  $\text{cred} \leftarrow \text{GrantCred}(\text{osk}, A)$  and adds the entry  $(\text{id}, A, \text{cred})$  to  $\mathcal{Q}$ . If  $\text{id} \in \mathcal{C}$  the oracle returns  $\text{cred}$ .

$\mathcal{O}^{\text{ShowCred}}(i, f)$ : The credential showing oracle takes as input an index  $i$  and a policy  $f$ . If  $i > |\mathcal{Q}|$  the oracle returns  $\perp$ , else it parses  $\mathcal{Q}[i]$  as  $(\text{id}, A, \text{cred})$  and executes:

$$\langle \text{ShowCred}(\text{opk}, \text{cred}, f), \cdot \rangle$$

in interaction with the adversary.

$\mathcal{O}^{\text{DelShowCred}}(i, \text{tok})$ : The delegated credential showing oracle takes as input an index  $i$  and a verification token  $\text{tok}$ . If  $i > |\mathcal{Q}|$  the oracle returns  $\perp$ , else it parses  $\mathcal{Q}[i]$  as  $(\text{id}, A, \text{cred})$  and executes:

$$\langle \text{DelShowCred}(\text{opk}, \text{cred}, \text{tok}), \cdot \rangle$$

in interaction with the adversary.

$\mathcal{O}_b^{\text{Anon}}(i_0, i_1, f)$ : The inputs to the anonymity oracle are two indices  $(i_0, i_1)$  and a policy  $f$ . If  $i_0 > |\mathcal{Q}|$  or  $i_1 > |\mathcal{Q}|$  the oracle returns  $\perp$ , else it parses  $\mathcal{Q}[i_0]$  as  $(\text{id}_0, A_0, \text{cred}_0)$  and  $\mathcal{Q}[i_1]$  as  $(\text{id}_1, A_1, \text{cred}_1)$ . If  $f(A_0) \neq f(A_1)$ , the oracle returns  $\perp$ . Otherwise it runs:

$$\langle \text{ShowCred}(\text{opk}, \text{cred}_b, f), \cdot \rangle$$

in interaction with the adversary.

$\mathcal{O}_b^{\text{AnonDel}}(i_0, i_1, \text{tok})$ : The inputs to the delegation anonymity oracle are two indices  $(i_0, i_1)$  and a verification token  $\text{tok}$ . If  $i_0 > |\mathcal{Q}|$  or  $i_1 > |\mathcal{Q}|$  the oracle returns  $\perp$ , else it parses  $\mathcal{Q}[i_0]$  as  $(\text{id}_0, A_0, \text{cred}_0)$  and  $\mathcal{Q}[i_1]$  as  $(\text{id}_1, A_1, \text{cred}_1)$ . If there exists an entry of the form  $(f, \text{tok}) \in \mathcal{L}$ , the oracle checks whether  $f(A_0) \neq f(A_1)$  and returns  $\perp$  if this is the case. Otherwise it runs:

$$\langle \text{DelShowCred}(\text{opk}, \text{cred}_b, \text{tok}), \cdot \rangle$$

in interaction with the adversary.

$\mathcal{O}_b^{\text{PH}}(f_0, f_1)$ : The policy-hiding oracle takes as input two challenge policies  $f_0$  and  $f_1$ . If there exists an entry  $(\text{id}, A, \text{cred}) \in \mathcal{Q}$  such that  $\text{id} \in \mathcal{C}$  and  $f_0(A) \neq f_1(A)$ , the oracles returns  $\perp$ . Otherwise it returns the token  $\text{tok}_b \leftarrow \text{Del}(\text{osk}, f_b)$ . From this moment on, the queries to  $\mathcal{O}^{\text{Corrupt}}$  are restricted to inputs  $\text{id}$  such that for all entries  $(\text{id}, A_i, \text{cred}_i) \in \mathcal{Q}$  it holds that  $f_0(A_i) = f_1(A_i)$ . The

$\text{ExpUnf}_{\text{FC},\mathcal{A}}(1^\lambda)$ <hr/> $(\text{opk}, \text{osk}) \leftarrow \text{CKGen}(1^\lambda)$ $\{f, \text{tok}\} \leftarrow \mathcal{A}^{\mathcal{O}_{\text{osk}}}(\text{opk}).$ <p><b>if</b> <math>f</math> <b>then</b></p> <p style="padding-left: 20px;"><b>if</b> <math>\forall (\text{id}, \mathbf{A}, \cdot) \in \mathcal{Q}</math> s.t. <math>\text{id} \in \mathcal{C} : f(\mathbf{A}) = 0</math> <b>then</b></p> <p style="padding-left: 40px;"><b>return</b> <math>\langle \mathbf{A}, \text{VrfyCred}(\text{opk}, f) \rangle</math></p> <p><b>if</b> <math>\text{tok}</math></p> <p style="padding-left: 20px;"><b>if</b> <math>\forall (f_{\text{tok}}, \text{tok}) \in \mathcal{L}, \forall (\text{id}, \mathbf{A}, \cdot) \in \mathcal{Q}</math> s.t. <math>\text{id} \in \mathcal{C} :</math></p> <p style="padding-left: 40px;"><math>f_{\text{tok}}(\mathbf{A}) = 0</math> <b>then</b></p> <p style="padding-left: 60px;"><b>return</b> <math>\langle \mathbf{A}, \text{DelVrfyCred}(\text{opk}, \text{tok}) \rangle</math></p> <p><b>return</b> 0</p>
--

Fig. 1. Game for unforgeability.

same restriction is applied for the oracle  $\mathcal{O}^{\text{DelShowCred}}$  when queried on  $\text{tok}_b$ .

**UNFORGEABILITY.** The notion of unforgeability captures the fact that any adversary cannot fool the verifier into accepting a credential for a policy that he does not satisfy. Intuitively, an adversary wins the unforgeability experiment if he is able to convince an honest verifier that he satisfies a certain policy while not holding an appropriate credential. We allow the adversary to specify any policy that no corrupted client can satisfy. The adversary can also win the game by fooling a designated verifier; in that case we check that no corrupted client has a valid attribute set only if the token was the result of a query of the adversary to the oracle  $\mathcal{O}_{\text{osk}}^{\text{Del}}$ . If the verification token is fresh, i.e., not an output of  $\mathcal{O}_{\text{osk}}^{\text{Del}}$ , we do not run any additional check.

**Definition 3.** A functional credential scheme FC is unforgeable if, for all PPT adversaries  $\mathcal{A}$  having access to the oracles  $\mathcal{O}_{\text{osk}} := (\mathcal{O}^{\text{User}}, \mathcal{O}^{\text{Corrupt}}, \mathcal{O}_{\text{osk}}^{\text{Del}}, \mathcal{O}_{\text{osk}}^{\text{GrantCred}}, \mathcal{O}^{\text{ShowCred}}, \mathcal{O}^{\text{DelShowCred}})$ , there exists a negligible function  $\text{negl}(\lambda)$  such that

$$\Pr[\text{ExpUnf}_{\text{FC},\mathcal{A}}(1^\lambda) = 1] \leq \text{negl}(\lambda),$$

where  $\text{ExpUnf}_{\text{FC},\mathcal{A}}(1^\lambda)$  is defined in Figure 1.

**ANONYMITY.** The definition of anonymity models the fact that a corrupted verifier cannot distinguish between any two users with different credentials, as long as they both satisfy or not satisfy the policy that they are tested against. The essence of the game is captured by the oracles  $\mathcal{O}_b^{\text{Anon}}$  and  $\mathcal{O}_b^{\text{AnonDel}}$ . Depending on the internal bit  $b$ , the oracles impersonate either one of the two input credential owners in the verification algorithm, on some

$\text{ExpAno}_{\text{FC},\mathcal{A}}^b(1^\lambda)$ <hr/> $(\text{opk}, \text{osk}) \leftarrow \text{CKGen}(1^\lambda)$ $b' \leftarrow \mathcal{A}^{\mathcal{O}_{b,\text{osk}}^{\text{ExpAno}}}(\text{opk})$ <p><b>return</b> <math>(b = b')</math></p>
--

Fig. 2. Game for anonymity

$\text{ExpPH}_{\text{FC},\mathcal{A}}^b(1^\lambda)$ <hr/> $(\text{opk}, \text{osk}) \leftarrow \text{CKGen}(1^\lambda)$ $b' \leftarrow \mathcal{A}^{\mathcal{O}_{b,\text{osk}}^{\text{ExpPH}}}(\text{opk})$ <p><b>return</b> <math>(b = b')</math></p>
--

Fig. 3. Game for policy-hiding

policy chosen by the adversary. In both cases, to make the game non-trivial, we must impose the restriction that the policy (or the policy associated with the token) is either satisfied or not by both credentials. We stress that the oracles can be queried adaptively and polynomially many times on arbitrary policies (or tokens) and pairs of users. This implies that our definition captures the notion of unlinkability: Any two credentials that satisfy a certain policy (or that both do not satisfy it) are indistinguishable to the eyes of the attacker.

**Definition 4.** A functional credential scheme FC is anonymous if, for all PPT adversaries  $\mathcal{A}$  having access to oracles  $\mathcal{O}_{b,\text{osk}}^{\text{ExpAno}} := (\mathcal{O}^{\text{User}}, \mathcal{O}^{\text{Corrupt}}, \mathcal{O}_{\text{osk}}^{\text{Del}}, \mathcal{O}_{\text{osk}}^{\text{GrantCred}}, \mathcal{O}^{\text{ShowCred}}, \mathcal{O}^{\text{DelShowCred}}, \mathcal{O}_b^{\text{Anon}}, \mathcal{O}_b^{\text{AnonDel}})$ , there exists a negligible function  $\text{negl}(\lambda)$  such that

$$\left| \Pr[\text{ExpAno}_{\text{FC},\mathcal{A}}^0(1^\lambda) = 1] - \Pr[\text{ExpAno}_{\text{FC},\mathcal{A}}^1(1^\lambda) = 1] \right| \leq \text{negl}(\lambda),$$

where the two experiments are defined in Figure 2.

**POLICY-HIDING.** The policy-hiding property describes the fact that each honestly issued verification token does not reveal any information about the policy that it is encoding beyond the validity of the credentials possessed by the adversary. The core of the security is modeled by the oracle  $\mathcal{O}_b^{\text{PH}}$  that, on input two policies by the adversary, returns the verification token for either one of the two depending on some internal coin  $b$ . To avoid trivial attacks we must ensure that clients corrupted by the adversary do not have any credential that have different access with respect to the two policies. Also the queries to the oracle  $\mathcal{O}^{\text{DelShowCred}}$  are restricted to those credentials having the same output on both of the challenge policies.

**Definition 5.** A functional credential scheme FC is policy-hiding if, for all PPT adversaries  $\mathcal{A}$  having access to oracles  $\mathcal{O}_{b,\text{osk}}^{\text{ExpPH}} := (\mathcal{O}^{\text{User}}, \mathcal{O}^{\text{Corrupt}}, \mathcal{O}_{\text{osk}}^{\text{Del}}, \mathcal{O}_{\text{osk}}^{\text{GrantCred}}, \mathcal{O}^{\text{ShowCred}}, \mathcal{O}^{\text{DelShowCred}}, \mathcal{O}_b^{\text{PH}})$ , there exists a negligible

function  $\text{negl}(\lambda)$  such that

$$\left| \Pr [\text{ExpPH}_{\text{FC},\mathcal{A}}^0(1^\lambda) = 1] - \Pr [\text{ExpPH}_{\text{FC},\mathcal{A}}^1(1^\lambda) = 1] \right| \leq \text{negl}(\lambda),$$

where the two experiments are defined in Figure 3.

## 2.3 Discussion

**CREDENTIAL DELEGATION.** Our model can be easily adapted to support delegation of credentials by defining the appropriate interface and providing the adversary with the corresponding oracle in the security games. For ease of explanation we, however, omit this functionality in our formal description.

**UNCONDITIONAL ANONYMITY.** We shall note that our definitions, in particular the anonymity one, require a trusted setup of the parameters of the issuer. This implies that our security definitions guarantee anonymity only under the assumption of a trustworthy issuer that distributes the credentials. The reason behind this choice is that unconditional anonymity in the presence of delegated verifier is, according to our definition, impossible to achieve. In fact, it appears that the policy-hiding property is in an inherent conflict with our goal: An adversary could adaptively generate tokens for arbitrarily restrictive policies and test the credential of a user against them, eventually uniquely identifying the carrier of a credential. Therefore, a scheme satisfying anonymity in the presence of an untrustworthy issuer must not support delegation of verification with policy-hiding. Due to the novelty of this feature, our work sacrifices the unconditional anonymity in favor of the delegation of verification feature. For completeness, we also outline how to modify our construction in order to achieve full anonymity in Section 3.4.

**REVOCAION.** As identified in the work of Camenisch et al. [15], an important feature to handle the dynamic nature of the set of users of a credential system is the possibility of revoking the credentials of some parties. Although our model does not directly support such a functionality, we can leverage the expressiveness of functional credentials in order to achieve it. The basic idea is to handle the revocation at the policy level: Since the verification algorithm can test credentials against any polynomial-time computable predicate, it is enough to add a conjunctive clause to the policy that prevents all the credentials possessing a certain attribute from satisfying it. That is, given a certain attribute  $a$  to revoke, the verifier must update the verification policy  $f$  to

$f' = f \wedge \neg a$ . Whitelisting can be done analogously, using disjunctive clauses instead. Note that, given the policy-hiding property of the functional credential scheme, one can even keep the set of revoked/whitelisted attributes private by issuing a verification token  $\text{tok}_{f'}$  for the updated policy.

## 3 Generic Construction

In this section we introduce our generic construction.

### 3.1 Preliminaries

Here we recall the definitions of our cryptographic building blocks.

**PREDICATE ENCRYPTION SCHEME.** A predicate encryption scheme allows one to embed attributes into the ciphertext and to encode arbitrary polynomial predicates in the decryption keys. We use the following notation:  $\Sigma$  denotes an arbitrary attribute domain and  $\mathcal{F}$  the family of polynomially computable predicates over  $\Sigma$ , which may depend on the security parameter  $\lambda$  and/or on the public parameters of the scheme. Our definition follows almost verbatim from the work of Katz et al. [34].

**Definition 6.** A predicate encryption scheme  $\Pi$  for a class of predicates  $\mathcal{F}$  over the domain  $\Sigma$  consists of four PPT algorithms ( $\text{Setup}$ ,  $\text{KGen}$ ,  $\text{Enc}$ ,  $\text{Dec}$ ) such that:

$\text{Setup}(1^\lambda)$ . The setup algorithm outputs a master secret key  $\text{dk}$  and a master public key  $\text{pk}$ .

$\text{KGen}(\text{dk}, f)$ . The key generation algorithm takes as input the master secret key and a predicate  $f \in \mathcal{F}$ . It outputs a key  $\text{dk}_f$ .

$\text{Enc}(\text{pk}, \mathcal{I}, m)$ . The encryption algorithm takes as input the public key  $\text{pk}$ , a set of attributes  $\mathcal{I} \in \Sigma$ , and a message  $m$  in some associated message space. It returns a ciphertext  $c$ .

$\text{Dec}(\text{dk}_f, c)$ . The decryption algorithm takes as input a secret key  $\text{dk}_f$  and a ciphertext  $c$ . It outputs either a message  $m$  or a distinguished message  $\perp$ .

For correctness, we require that for all  $\lambda \in \mathbb{N}$ , all  $m \in \{0, 1\}^{\text{poly}(\lambda)}$ , all  $(\text{pk}, \text{dk}) \in \text{Setup}(1^\lambda)$ , all  $f \in \mathcal{F}$ , all  $\text{sk}_f \in \text{KGen}(\text{dk}, f)$ , and all  $\mathcal{I} \in \Sigma$ :

- If  $f(\mathcal{I}) = 1$  then  $\text{Dec}(\text{dk}_f, \text{Enc}(\text{pk}, \mathcal{I}, m)) = m$ .
- If  $f(\mathcal{I}) = 0$  then  $\text{Dec}(\text{dk}_f, \text{Enc}(\text{pk}, \mathcal{I}, m)) = \perp$  with all but negligible probability.

A predicate encryption achieves payload-hiding if the message of a ciphertext is hidden from the eyes of the users whose predicate does not satisfy the encoded set of attributes. The attribute-hiding notion additionally requires that the ciphertext conceals the corresponding set of attributes. In the following we recall the definition for the latter. Note that we consider the adaptive variant of it.

**Definition 7.** A predicate encryption scheme  $\Pi$  is attribute-hiding with respect to  $\mathcal{F}$  and  $\Sigma$  if for all PPT adversaries  $\mathcal{A}$ , there exists a negligible function  $\text{negl}(\lambda)$  such that:

$$\left| \Pr [\text{ExpAH}_{\Pi, \mathcal{A}}^0(1^\lambda) = 1] - \Pr [\text{ExpAH}_{\Pi, \mathcal{A}}^1(1^\lambda) = 1] \right| \leq \text{negl}(\lambda),$$

where  $\text{ExpAH}_{\Pi, \mathcal{A}}^b(1^\lambda)$  is defined as follows.

```

ExpAHΠ, Ab(1λ)
((dk, pk) ← Setup(1λ))
((m0, m1), (I0, I1)) ← AKGen(dk, ·)(pk)
Let Q denote the set of oracle query-answer pairs.
if ∀f s.t. (f, ·) ∈ Q : f(I0) = f(I1) and m0 = m1
  or ∀f s.t. (f, ·) ∈ Q : f(I0) = f(I1) = 0 then
  c ← Enc(pk, Ib, mb)
  b' ← AKGen*(dk, ·)(c)
  return b = b'
else return 0
    
```

Where  $\text{KGen}^*$ , on input some  $f$ , checks the same condition as above and returns  $\text{KGen}(\text{osk}, f)$  if it holds.

**ADDITIONAL PROPERTIES.** Let the message space of the scheme be a multiplicative cyclic group of prime order  $p$ . We additionally require the existence of an operator  $\otimes$  and an algorithm  $\text{ReRand}(\cdot, \cdot)$  such that for all key pairs  $(\text{pk}, \text{dk}) \in \text{Setup}(1^\lambda)$ , for all pairs of messages  $(m, n) \in \{0, 1\}^{2\text{poly}(\lambda)}$ , for all attributes  $\mathcal{I} \in \Sigma$ , for all  $c \in \text{Enc}(\text{pk}, \mathcal{I}, m)$ , it holds that  $\text{ReRand}(\text{pk}, c \otimes n) \approx \text{Enc}(\text{pk}, \mathcal{I}, m \cdot n)$ , where  $\approx$  denotes statistical indistinguishability.

**HIERARCHICAL PREDICATE ENCRYPTION.** Recent developments in the field of predicate encryption explored the possibility of a hierarchical delegation of predicates. The work of Okamoto and Takashima [39] first introduced the concept of hierarchical predicate encryption scheme, where possessors of capabilities for certain predicates can delegate to other users more restrictive capabilities. Here the property is formalized only for inner-product

predicates but one can extend it to more generic classes of functions. Loosely speaking, the owner of a key  $\text{dk}_f$ , for a certain predicate  $f$ , can delegate keys for any predicate  $f'$  such that, for all  $\mathcal{I} \in \Sigma$ ,  $f'(\mathcal{I}) = 1 \implies f(\mathcal{I}) = 1$ . Since our scheme does not primarily focus on delegation of credentials, we refrain from formally defining such a primitive and we only sketch how one can extend our construction using hierarchical predicate encryption.

**COMMITMENT SCHEME.** We recall the notion of non-interactive, equivocal, and extractable commitment schemes. These commitments fulfill the usual definitions of binding and hiding, but in addition they are equivocal and extractable, properties that were first introduced by Canetti and Fischlin in the context of universally composable commitments [23].

**Definition 8.** A non-interactive commitment scheme  $(\mathcal{P}, \mathcal{V})$  is a two-phase protocol between two PPT parties  $\mathcal{P}$  and  $\mathcal{V}$ , called the committer and the receiver, respectively, such that the following holds: in the first phase, the commitment phase, given the common reference string  $\text{crs}$ ,  $\mathcal{P}$  commits to a message  $m$  by computing a pair  $(\text{com}, \text{decom})$  and sending  $\text{com}$  to  $\mathcal{V}$ . In the second phase (the decommitment phase)  $\mathcal{P}$  reveals the key  $\text{decom}$  to  $\mathcal{V}$ .  $\mathcal{V}$  checks whether the decommitment key is valid; if not,  $\mathcal{V}$  outputs  $\perp$ , meaning that he rejects the decommitment from  $\mathcal{P}$ , otherwise  $\mathcal{V}$  can efficiently compute the message  $m$ .

For correctness, we require that for all  $\text{crs} \in \{0, 1\}^{\text{poly}(\lambda)}$ , for all  $m \in \{0, 1\}^{\text{poly}(\lambda)}$ , for all  $(\text{com}, \text{decom}) \in \mathcal{P}(\text{crs}, m)$  there exists a negligible function  $\text{negl}(\lambda)$  in the security parameter  $\lambda$  such that:

$$\Pr[m \leftarrow \mathcal{V}(\text{crs}, \text{com}, \text{decom})] \geq 1 - \text{negl}(\lambda).$$

A commitment scheme is *hiding* if any honestly generated commitment  $\text{com}$  reveals no information about the committed message  $m$ . Additionally, a commitment scheme is *binding* if there does not exist any efficient algorithm that is able to output two keys  $(\text{decom}_0, \text{decom}_1)$  for the same commitment  $\text{com}$  such that they open it to two different values. We do not formally define these two properties as they are implied by the following two (stronger) requirements. Loosely speaking, a commitment scheme is *equivocal* if one can sample a trapdoor common reference string from a distribution that is computationally indistinguishable from that of the original that allows the possessor of the trapdoor to cheat during the opening of a commitment. In fact, the trapdoor makes it possible for the prover to

convince the verifier that a previously computed commitment opens to an arbitrary message. Additionally, the *extractability* property guarantees that the possessor of the trapdoor can extract the committed value from any honestly generated commitment, without the opening.

**Definition 9.** A non-interactive commitment scheme  $(\mathcal{P}, \mathcal{V})$  is equivocable and extractable if there exists a tuple of PPT algorithms  $(\mathcal{E}, \mathcal{E}_{\text{Eq}}^0, \mathcal{E}_{\text{Eq}}^1, \mathcal{E}_{\text{Ext}})$  such that for all  $m \in \{0, 1\}^{\text{poly}(\lambda)}$ , for all  $(\text{crs}, \alpha) \in \mathcal{E}(1^\lambda)$ ,  $\text{com} \in \mathcal{E}_{\text{Eq}}^0(\text{crs}, \alpha)$ , and  $\text{decom} \in \mathcal{E}_{\text{Eq}}^1(\text{crs}, \alpha, \text{com}, m)$  it holds that:  $m \leftarrow \mathcal{V}(\text{crs}, \text{com}, \text{decom})$  and the families of random variables

$$\left\{ \begin{array}{l} \text{crs} \leftarrow \{0, 1\}^{\text{poly}(\lambda)}; (\text{com}, \text{decom}) \leftarrow \mathcal{P}(\text{crs}, m) : \\ (\text{crs}, \text{com}, \text{decom}) \end{array} \right\}$$

and

$$\left\{ \begin{array}{l} \{(\text{crs}, \alpha) \leftarrow \mathcal{E}(1^\lambda); \text{com} \leftarrow \mathcal{E}_{\text{Eq}}^0(\text{crs}, \alpha); \\ \text{decom} \leftarrow \mathcal{E}_{\text{Eq}}^1(\text{crs}, \alpha, \text{com}, m) : (\text{crs}, \text{com}, \text{decom}) \end{array} \right\}$$

are computationally indistinguishable. Additionally it must hold that there exists a negligible function  $\text{negl}(\lambda)$  such that for all algorithms  $\mathcal{P}^*$ , for all  $m \in \{0, 1\}^{\text{poly}(\lambda)}$ :

$$\left| \begin{array}{l} \Pr \left[ \begin{array}{l} (\text{crs}, \alpha) \leftarrow \mathcal{E}(1^\lambda); \\ (\text{com}, \text{decom}) \leftarrow \mathcal{P}^*(\text{crs}, m); \\ m' \leftarrow \mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}) : \\ \mathcal{V}(\text{crs}, \text{com}, \text{decom}) = m' \end{array} \right] - \\ \Pr \left[ \begin{array}{l} \text{crs} \leftarrow \{0, 1\}^{\text{poly}(\lambda)}; \\ (\text{com}, \text{decom}) \leftarrow \mathcal{P}^*(\text{crs}, m) : \\ \mathcal{V}(\text{crs}, \text{com}, \text{decom}) = m \end{array} \right] \end{array} \right| \leq \text{negl}(\lambda).$$

**DIGITAL SIGNATURE SCHEME.** We assume that the reader is familiar with the standard definition of existentially unforgeable [31] signature schemes  $\text{Sig} = (\text{SKGen}, \text{Sig}, \text{Vf})$  and omit the formal definition.

## 3.2 The Scheme

In the following we present our construction for a functional credential scheme FC. Our construction makes black-box usage of any predicate encryption scheme whose attribute space  $\Sigma$  is a set of binary strings of length polynomial in the security parameter and whose family of predicates  $\mathcal{F}$  is a collection of polynomial-time computable functions over  $\Sigma$ . We encode binary descriptions of policies (which belong to  $\Sigma$ ) in ciphertexts, and we define  $f$  as the predicate that evaluates

the circuit encoded in the ciphertext over some hard-coded set of attributes. Note that  $f$  is a polynomially computable function over  $\Sigma$  and it is therefore a valid predicate for a secret key of the scheme. Jumping ahead to our instantiation, we can avoid the costly dual function encoding by exploiting the structure of the underlying predicate encryption scheme. A concrete example of our encoding for a boolean formula is given in Section 4. Let  $\Pi = (\text{Setup}, \text{KGen}, \text{Enc}, \text{Dec})$  be a predicate encryption scheme, let  $(\mathcal{P}, \mathcal{V})$  be a non-interactive commitment scheme and let  $\text{Sig} = (\text{SKGen}, \text{Sig}, \text{Vf})$  be a digital signature scheme. The formal description of the construction is elaborated below.

$\text{CKGen}(1^\lambda)$ : The key generation algorithm runs  $(\text{dk}, \text{pk}) \leftarrow \text{Setup}(1^\lambda)$ ,  $(\text{sk}, \text{vk}) \leftarrow \text{SKGen}(1^\lambda)$ , and uniformly samples  $\text{crs} \leftarrow \{0, 1\}^{\text{poly}(\lambda)}$ . It returns  $\text{opk} = (\text{pk}, \text{vk}, \text{crs})$  and  $\text{osk} = (\text{dk}, \text{sk}, \text{opk})$ .

$\text{GrantCred}(\text{osk}, A)$ : The grant credential algorithm parses  $\text{osk}$  as  $(\text{dk}, \text{sk}, \text{opk})$  and defines the predicate  $f_A(g)$  that takes as input the description of some predicate  $g$  and returns  $g(A)$ . Then the algorithm executes  $\text{dk}_{f_A} \leftarrow \text{KGen}(\text{dk}, f_A)$  and returns  $\text{cred} = \text{dk}_{f_A}$ .

$\text{Del}(\text{osk}, f)$ : The delegation algorithm parses  $\text{osk}$  as  $(\text{dk}, \text{sk}, \text{opk})$  and  $\text{opk}$  as  $(\text{pk}, \text{vk}, \text{crs})$ . It generates  $c \leftarrow \text{Enc}(\text{pk}, f, 1)$ , where  $f$  is a description of the input predicate. The algorithm runs  $\sigma \leftarrow \text{Sig}(\text{sk}, c)$  and returns  $\text{tok} = (c, \sigma)$ .

$(\text{ShowCred}(\text{opk}, \text{cred}, f), \text{VrfyCred}(\text{opk}, f))$ : The idea behind the interactive protocol is that the prover shows the possession of a certain credential  $\text{cred}$  by decrypting a ciphertext  $c \leftarrow \text{Enc}(\text{pk}, f, s; t)$  that encodes the desired policy and is uniformly distributed in the appropriate domain. To make sure that the ciphertext was honestly computed, the prover first commits to the result of the decryption and let the verifier disclose the randomness  $t$  that was used to compute the ciphertext. If indeed the ciphertext was honestly generated, then the prover sends the decommitment information, otherwise it aborts the protocol. The verifier accepts if the decommitment information reconstructs to the plaintext  $s$  of the original ciphertext. A similar technique based on the commitment of the randomness  $r$  is used to ensure that the ciphertext is randomly generated. The full protocol is depicted in Figure 4.

$(\text{DelShowCred}(\text{opk}, \text{cred}, \text{tok}), \text{DelVrfyCred}(\text{opk}, \text{tok}))$ : The interactive algorithm for delegated verification closely resembles the one above, with the difference that the ciphertext to be decrypted is not generated by the verifier but is a rerandomization of a token received from the

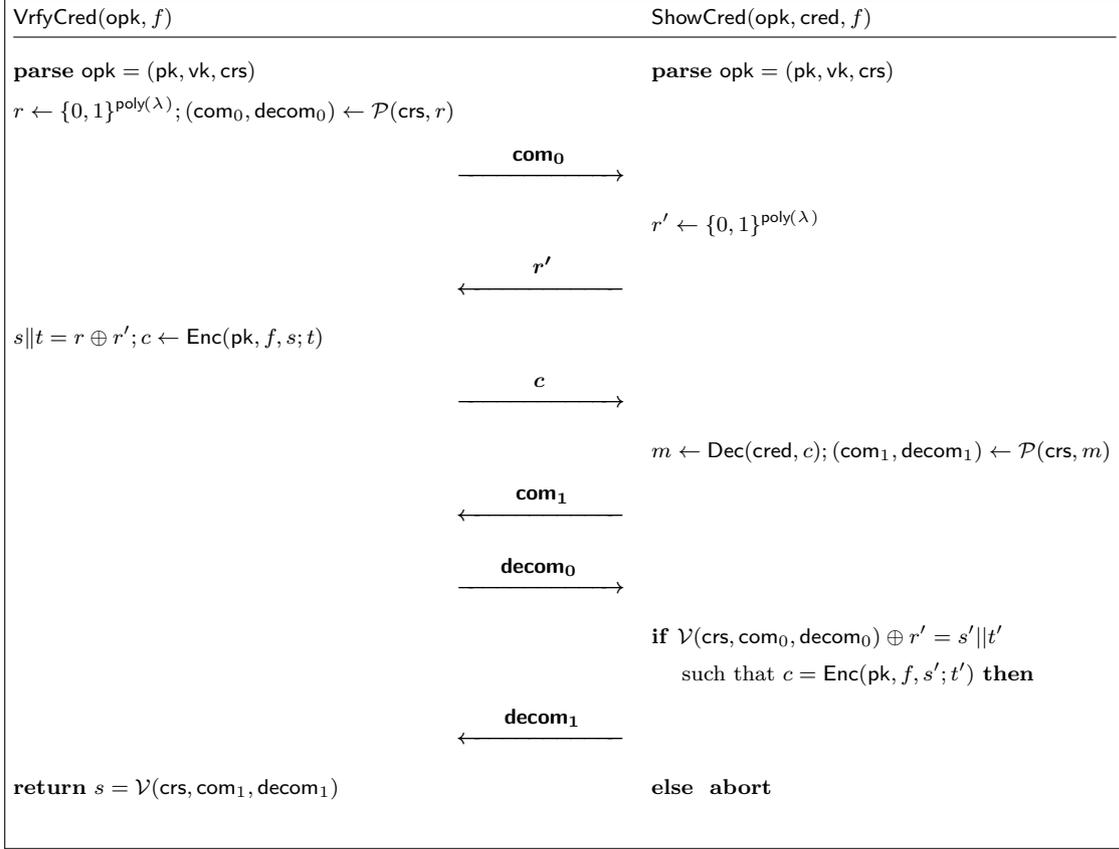


Fig. 4. Verification protocol.

issuer. The token is signed with the key of the issuer to vouch for its authenticity. We elaborate a pictorial description of the protocol in Figure 5.

### 3.3 Security Analysis

In the following we formally state the security of our construction as defined in Section 3.

**Theorem 1.** *Let  $\Pi = (\text{Setup}, \text{KGen}, \text{Enc}, \text{Dec})$  be a homomorphic attribute-hiding predicate encryption scheme,  $\text{Sig} = (\text{SKGen}, \text{Sig}, \text{Vf})$  be an existentially unforgeable digital signature scheme, and  $(\mathcal{P}, \mathcal{V})$  be an equivocable and extractable commitment scheme, then FC is an unforgeable, anonymous, and policy-hiding functional credential scheme.*

Security is defined as the existence of a simulator for each corrupted party. Note that, as opposed to generic proofs of knowledge, our simulator cannot extract the witness (the secret key for a certain predicate) of a successful run. However, the payload-hiding of the predicate encryption scheme guarantees that the possession

of the key is necessary to decrypt a given ciphertext. It follows that we can prove the scheme secure without additional assumptions. For a formal treatment of the security proof we refer the reader to Appendix A.

### 3.4 Extensions

**UNCONDITIONAL ANONYMITY.** We now show how to modify our construction to achieve anonymity in the presence of a malicious issuer, in the common reference string model: The main observation is that the verification protocol remains anonymous even if the master secret key of the predicate encryption scheme is leaked, i.e., the anonymity of our construction does not rely on the security of the predicate encryption scheme (see Appendix A for details). Therefore, in order to obtain full anonymity, our construction must guarantee that the public parameters and the credentials issued by any issuer are constructed accordingly to the specifications of the algorithms. This can be easily realized by attaching non-interactive zero-knowledge proofs of well-formedness to the public key and to each credential,

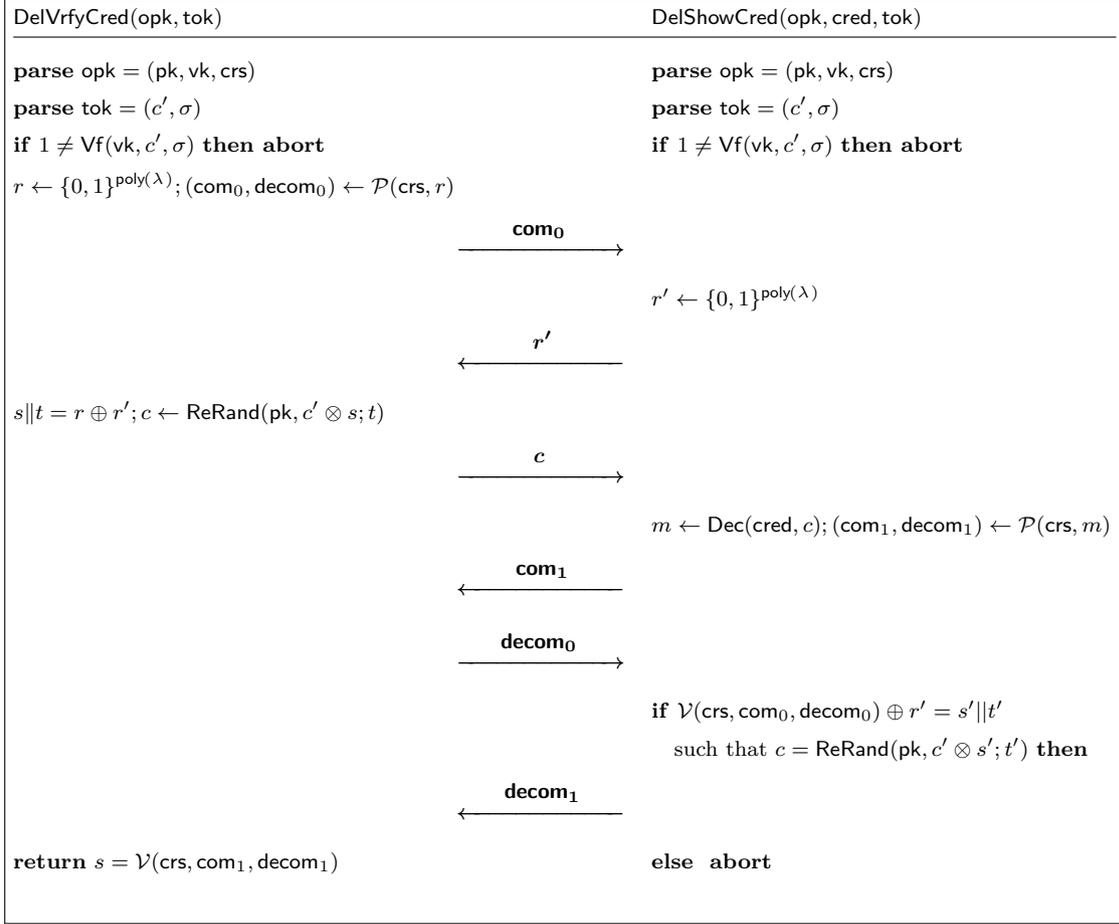


Fig. 5. Delegated Verification protocol.

which can be efficiently checked by any user. Proofs in the common reference string model can be instantiated using the Groth-Sahai framework [32]. Note that this solution introduces a significant computational overhead in the system initialization and in the credential issuing (the issuer must compute the proofs and each user must verify them), but it does not affect the verification algorithm, arguably the most frequently executed operation of a credential system. As we discussed before, unconditional anonymity cannot be achieved together with a policy-hiding delegation of verification, therefore the credential system must not support delegation of verification tokens. This means that a payload-hiding predicate encryption scheme would suffice to guarantee the security of our protocol, allowing us to instantiate our construction with a broader class of schemes [7, 43].

**DELEGATION OF CREDENTIALS.** Hierarchical predicate encryption provides a very natural way to delegate credentials: A credential for a set of attributes  $A$  corresponds to a decryption key  $\text{dk}_{f_A}$ , therefore the owner of  $\text{dk}_{f_A}$  can delegate a credential for a set  $A'$  by gener-

ating the corresponding secret key  $\text{dk}_{f_{A'}}$ . Note that one can only delegate keys for less powerful predicates, i.e., it must hold that  $f_{A'}(g) = 1 \implies f_A(g) = 1$ . Since  $f_A$  evaluates some input function  $g$  on  $A$ , we have that  $A' \subseteq A$ . Clearly the same holds for the specific case of inner-product schemes.

## 4 Experimental Evaluation

In this section we show how to instantiate our generic construction and we provide the reader with an experimental evaluation of our scheme.

**PREDICATE ENCODING.** Recall that our protocol relies generically on a predicate encryption scheme and known instances of predicate encryption support only inner-product predicates. Specifically, the schemes have an attribute domain  $\Sigma = \mathbb{Z}_p^n$  and a corresponding class of predicates  $\mathcal{F} = \{f_{\vec{y}} | \vec{y} \in \mathbb{Z}_p^n\}$ , where  $f_{\vec{y}} : \mathbb{Z}_p^n \rightarrow \{0, 1\}$  is

defined as

$$f_{\vec{y}}(\vec{x}) = \begin{cases} 1 & \text{if } \langle \vec{x}, \vec{y} \rangle \equiv 0, \text{ and,} \\ 0 & \text{otherwise.} \end{cases}$$

Using the generic transformation described by Katz et al. [34], we can encode boolean formulae in their  $k$ -CNF (for some fixed  $k$ ) as inner product evaluations. As an example, consider the policy  $X > 18$ . Let  $(X_1, \dots, X_m)$  and  $(L_1, \dots, L_m)$  be the bit representations (starting from the most significant bit) of  $X$  and 18, respectively. Such a policy admits an efficient representation as a DNF formula:

$$\bigvee_{i=1}^m \left( \bigwedge_{j=1}^{i-1} (X_j \oplus \overline{L_j}) \wedge (X_i \oplus L_i) \wedge X_i \right).$$

Note that such a formula can be converted into an equivalent CNF using De Morgan’s law, which may cause an exponential expansion of the formula. For this reason we consider only a constant amount of boolean variables. In our encoding each boolean variable  $B$  is associated with a unique variable  $b \in \mathbb{Z}_p$  that can take one of the two values:  $b^0$  if  $B = 0$  and  $b^1$  if  $B = 1$ , so in this context checking  $B = 1$  means checking whether  $b = b^1$ . Consider the following multivariate polynomial

$$P(x_1, \dots, x_m) = \prod_{i=1}^m \left( \sum_{j=1}^{i-1} r_{i,j} (x_j - e_j^{\overline{L_j}}) + s_i (x_i - e_i^{L_i}) + t_i (x_i - e_i^1) \right)$$

where the terms  $\{e_i^0, e_i^1\}_{i \in [m]}$  correspond to unique  $\mathbb{Z}_p$  elements and the vectors  $(\vec{r}, \vec{s}, \vec{t})$  are randomly chosen by the encoder. Recall that in our scheme policies are encoded at encryption time so the randomness is taken from the random coins of the encryption algorithm. We stress that boolean negations are handled associating an independent element of  $\mathbb{Z}_p$ , e.g.,  $\overline{L_j}$  is encoded as  $e_j^{\overline{L_j}}$ . Expanding the equation above and assigning a new variable to each monomial, we obtain a polynomial  $P(\vec{x})$  that evaluates to 0 only if the vector  $\vec{x}$  is a satisfying assignment for the formula as described above. Note that the evaluation of  $P$  corresponds to the computation of an inner product over vectors in  $\mathbb{Z}_p^n$ , where any  $n \geq 2^m$  suffices to evaluate  $P$ . The vector size  $n$  imposes a bound on the size of the boolean formula that can be encoded in each ciphertext, due to the exponential blowup of the disjunctive clauses. Therefore one must show that the scheme remains efficient for large values of  $n$ , in order to encode meaningful policies.

## 4.1 Implementation

In the following we elaborate on the details of the implementation and we present an experimental evaluation of our construction.

**CRYPTOGRAPHIC INSTANTIATIONS.** In our instantiation, we use the adaptively secure inner-product encryption scheme of Okamoto and Takashima [40] as a predicate encryption, and Schnorr’s signatures [42]. Although equivocal and extractable commitment schemes are known to exist in the common reference string model [27], for efficiency reasons we instantiate our commitment scheme in the random oracle model [6] with the well known construction  $H(m, r)$ . Throughout the following paragraphs, we fix the security parameter to be  $\lambda = 128$ . The random oracle is implemented using SHA-256. Note that the length of the messages that we commit to is proportional to the security parameter, but fixed throughout the execution of our protocol.

**EXPERIMENTS.** We implemented our cryptographic construction in C++ using the PBC library [36] with the default curve, which is  $y^2 = x^3 - x$  over the field  $\mathbb{F}_p$  for some prime  $p \equiv 3 \pmod{4}$ , on commodity hardware (i7 processor with 8 threads). We use OpenMP [26] for parallelization. The running times of our scheme are largely dominated by the operations relative to the predicate encryption scheme. We evaluated the computational efficiency of the algorithms of our construction in relation to the length  $n$  of the vector of the inner-product predicate encryption scheme. Recall that the value of  $n$  induces an upper bound on the size of the policy that one can encode in our construction. Our prototype implementation neglects one-time operations such as setup and credential generation and focuses on the recurrent interactive algorithms for showing and verifying the validity of a credential. We report the running times of the verification and the showing algorithms in Figure 6 and Figure 7, respectively. Each measurement is the average of 100 trials with standard deviation error bars. The magnitude of the standard deviation is explained by the fact that the runtime of the algorithms is largely dominated by the operations over the predicate encryption scheme (two encryptions and one decryption). Since a credential corresponds to a secret key of the scheme of Okamoto and Takashima [40], the size of a credential is constant and corresponds to 11 elements of the corresponding cyclic group. Note that the computation performed in the interactive protocol of delegated verification is the same as the standard verification (except for the verification of a signature from both parties) which

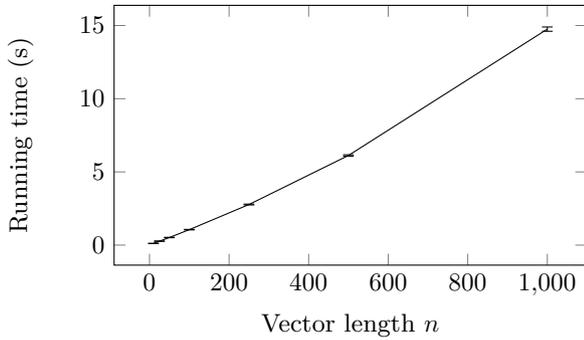


Fig. 6. Running times of VrfyCred.

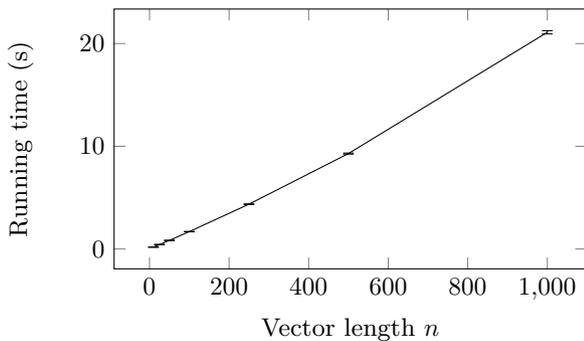


Fig. 7. Running times of ShowCred.

is why we do not explicitly state the numbers here. Remarkably, in our parallelized implementation, the total computation of the interactive verification protocol for  $n = 10^3$  is on the order of seconds (approximately 21).

FUNCTIONAL CREDENTIALS IN ORAM. To demonstrate the general applicability of our functional credential construction, we deploy it in the setting of Group ORAM (GORAM) [37], a state-of-the-art multi-client oblivious outsourced storage in which access control is enforced via predicate encryption and zero-knowledge proofs. More precisely, the authors encode the access control policy for each entry in a predicate encryption ciphertext and distribute a key to each client according to her permissions. Upon modification of a certain entry, the client must show a proof of knowledge of a key that allows her to successfully decrypt the associated predicate encryption ciphertext. Note that here the length  $n$  of the vector space supported by the predicate encryption bounds from above the total number of users of the system. Our delegated verification protocol can be plugged in GORAM in order to provide the same functionality while significantly boosting the performance. We compare in Table 2 the time associated with the computation and verification of a zero-knowledge proof (as implemented in GORAM) over a

Number of clients	3	4	5	6	7	8	9	10
GORAM (Server)	1002	1282	1489	1845	2111	2388	2793	3113
GORAM (Client)	1719	2184	2675	3136	3620	4121	4753	5239
This work (Server)	44	56	66	74	86	98	105	115
This work (Client)	65	83	100	114	133	152	164	181

Table 2. Efficiency improvements over GORAM [37] in terms of computation time for showing (Client) and verifying (Server) a credential. All of the measurements are expressed in milliseconds.

predicate encryption scheme with our functional credential scheme, parametrized by the number of clients  $n$  supported by the system. We stress that GORAM implements the scheme of Katz et al. [34] and therefore achieves a weaker notion of security compared to our scheme. Nevertheless, we observe that our approach substantially increases the scalability of the system with respect to the number of users, resulting in an efficiency improvement of one order of magnitude for 10 clients (5 s vs 0.2 s on the client-side and 3 s vs 0.1 s on the server side). We expect the gap to increase together with the number of clients.

## 5 Conclusions

This work introduces functional credentials. We formalize anonymous functional credentials and their security notions in terms of cryptographic games. We propose a general realization in the standard model with an efficient instantiation in the random oracle model. Our performance evaluations show that even for large parameters the computations for showing and verifying credentials is feasible for practical applications.

## Acknowledgments

This research is based upon work supported by the German research foundation (DFG) through the collaborative research center 1223, by the German Federal Ministry of Education and Research (BMBF) through the project PROMISE (16KIS0763), and by the state of Bavaria at the Nuremberg Campus of Technology (NCT). NCT is a research cooperation between the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and the Technische Hochschule Nürnberg Georg Simon Ohm (THN). This work was partly supported by the FFG Bridge 1 project pDLART, the ERC consolidator grant Browsec, and the FWF Doctoral College LogiCS.

## References

- [1] T. Acar and L. Nguyen. Revocation for delegatable anonymous credentials. In D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 423–440, Taormina, Italy, Mar. 6–9, 2011. Springer, Heidelberg, Germany.
- [2] M. H. Au, A. Kapadia, and W. Susilo. BLACR: TTP-free blacklistable anonymous credentials with reputation. In *NDSS 2012*, San Diego, California, USA, Feb. 5–8, 2012. The Internet Society.
- [3] M. Backes, S. Meiser, and D. Schröder. Delegatable functional signatures. In C.-M. Cheng, K.-M. Chung, G. Persiano, and B.-Y. Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 357–386, Taipei, Taiwan, Mar. 6–9, 2016. Springer, Heidelberg, Germany.
- [4] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, and H. Shacham. Randomizable proofs and delegatable anonymous credentials. In S. Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 108–125, Santa Barbara, CA, USA, Aug. 16–20, 2009. Springer, Heidelberg, Germany.
- [5] M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya. P-signatures and noninteractive anonymous credentials. In R. Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 356–374, San Francisco, CA, USA, Mar. 19–21, 2008. Springer, Heidelberg, Germany.
- [6] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73, Fairfax, Virginia, USA, Nov. 3–5, 1993. ACM Press.
- [7] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *2007 IEEE Symposium on Security and Privacy*, pages 321–334, Oakland, California, USA, May 20–23, 2007. IEEE Computer Society Press.
- [8] E. Boyle, S. Goldwasser, and I. Ivan. Functional signatures and pseudorandom functions. In H. Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 501–519, Buenos Aires, Argentina, Mar. 26–28, 2014. Springer, Heidelberg, Germany.
- [9] R. W. Bradshaw, J. E. Holt, and K. E. Seamons. Concealing complex policies with hidden credentials. In V. Atluri, B. Pfitzmann, and P. McDaniel, editors, *ACM CCS 04*, pages 146–157, Washington D.C., USA, Oct. 25–29, 2004. ACM Press.
- [10] J. Brandt, I. Damgård, P. Landrock, and T. P. Pedersen. Zero-knowledge authentication scheme with secret key exchange (extended abstract) (rump session). In S. Goldwasser, editor, *CRYPTO '88*, volume 403 of *LNCS*, pages 583–588, Santa Barbara, CA, USA, Aug. 21–25, 1990. Springer, Heidelberg, Germany.
- [11] J. Camenisch, M. Dubovitskaya, R. R. Enderlein, and G. Neven. Oblivious transfer with hidden access control from attribute-based encryption. In I. Visconti and R. D. Prisco, editors, *SCN 12*, volume 7485 of *LNCS*, pages 559–579, Amalfi, Italy, Sept. 5–7, 2012. Springer, Heidelberg, Germany.
- [12] J. Camenisch, M. Dubovitskaya, K. Haralambiev, and M. Kohlweiss. Composable and modular anonymous credentials: Definitions and practical constructions. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 262–288, Auckland, New Zealand, Nov. 30 – Dec. 3, 2015. Springer, Heidelberg, Germany.
- [13] J. Camenisch, M. Dubovitskaya, G. Neven, and G. M. Zaverucha. Oblivious transfer with hidden access control policies. In D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 192–209, Taormina, Italy, Mar. 6–9, 2011. Springer, Heidelberg, Germany.
- [14] J. Camenisch, M. Kohlweiss, and C. Soriente. Solving revocation with efficient update of anonymous credentials. In J. A. Garay and R. D. Prisco, editors, *SCN 10*, volume 6280 of *LNCS*, pages 454–471, Amalfi, Italy, Sept. 13–15, 2010. Springer, Heidelberg, Germany.
- [15] J. Camenisch, S. Krenn, A. Lehmann, G. L. Mikkelsen, G. Neven, and M. Ø. Pedersen. Formal treatment of privacy-enhancing credential systems. *Cryptology ePrint Archive*, Report 2014/708, 2014. <http://eprint.iacr.org/2014/708>.
- [16] J. Camenisch and A. Lehmann. (Un)linkable pseudonyms for governmental databases. In I. Ray, N. Li, and C. Kruegel, editors, *ACM CCS 15*, pages 1467–1479, Denver, CO, USA, Oct. 12–16, 2015. ACM Press.
- [17] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In B. Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118, Innsbruck, Austria, May 6–10, 2001. Springer, Heidelberg, Germany.
- [18] J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In S. Cimato, C. Galdi, and G. Persiano, editors, *SCN 02*, volume 2576 of *LNCS*, pages 268–289, Amalfi, Italy, Sept. 12–13, 2003. Springer, Heidelberg, Germany.
- [19] J. Camenisch and A. Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 56–72, Santa Barbara, CA, USA, Aug. 15–19, 2004. Springer, Heidelberg, Germany.
- [20] J. Camenisch and E. Van Herreweghen. Design and implementation of the idemix anonymous credential system. In V. Atluri, editor, *ACM CCS 02*, pages 21–30, Washington D.C., USA, Nov. 18–22, 2002. ACM Press.
- [21] S. Canard and R. Lescuyer. Anonymous credentials from (indexed) aggregate signatures. In *DIM'11, Proceedings of the 2013 ACM Workshop on Digital Identity Management, Chicago, IL, USA - October 21, 2011*, 2011.
- [22] S. Canard and R. Lescuyer. Protecting privacy by sanitizing personal data: a new approach to anonymous credentials. In K. Chen, Q. Xie, W. Qiu, N. Li, and W.-G. Tzeng, editors, *ASIACCS 13*, pages 381–392, Hangzhou, China, May 8–10, 2013. ACM Press.
- [23] R. Canetti and M. Fischlin. Universally composable commitments. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 19–40, Santa Barbara, CA, USA, Aug. 19–23, 2001. Springer, Heidelberg, Germany.
- [24] D. Chaum. Verification by anonymous monitors. In A. Gersho, editor, *CRYPTO '81*, volume ECE Report 82-04, pages 138–139, Santa Barbara, CA, USA, 1981. U.C. Santa Bar-

- bara, Dept. of Elec. and Computer Eng.
- [25] D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. L. Rivest, and A. T. Sherman, editors, *CRYPTO'82*, pages 199–203, Santa Barbara, CA, USA, 1982. Plenum Press, New York, USA.
- [26] L. Dagum and R. Menon. Openmp: An industry-standard api for shared-memory programming. *IEEE Comput. Sci. Eng.*, 5(1):46–55, Jan. 1998.
- [27] I. Damgård and J. B. Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 581–596, Santa Barbara, CA, USA, Aug. 18–22, 2002. Springer, Heidelberg, Germany.
- [28] K. B. Frikken, J. Li, and M. J. Atallah. Trust negotiation with hidden credentials, hidden policies, and policy cycles. In *NDSS 2006*, San Diego, California, USA, Feb. 2–3, 2006. The Internet Society.
- [29] G. Fuchsbaauer, C. Hanser, and D. Slamanig. Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. Cryptology ePrint Archive, Report 2014/944, 2014. <http://eprint.iacr.org/2014/944>.
- [30] C. Gentry. Fully homomorphic encryption using ideal lattices. In M. Mitzenmacher, editor, *41st ACM STOC*, pages 169–178, Bethesda, Maryland, USA, May 31 – June 2, 2009. ACM Press.
- [31] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, Apr. 1988.
- [32] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In N. P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432, Istanbul, Turkey, Apr. 13–17, 2008. Springer, Heidelberg, Germany.
- [33] C. Hanser and D. Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In P. Sarkar and T. Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 491–511, Kaoshiung, Taiwan, R.O.C., Dec. 7–11, 2014. Springer, Heidelberg, Germany.
- [34] J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In N. P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 146–162, Istanbul, Turkey, Apr. 13–17, 2008. Springer, Heidelberg, Germany.
- [35] V. Kolesnikov, H. Krawczyk, Y. Lindell, A. J. Malozemoff, and T. Rabin. Attribute-based key exchange with general policies. In *ACM CCS 16*, pages 1451–1463. ACM Press, 2016.
- [36] B. Lynn. The pairing-based cryptography (pbc) library, 2010.
- [37] M. Maffei, G. Malavolta, M. Reinert, and D. Schröder. Privacy and access control for outsourced personal records. In *2015 IEEE Symposium on Security and Privacy*, pages 341–358, San Jose, California, USA, May 17–21, 2015. IEEE Computer Society Press.
- [38] T. Nishide, K. Yoneyama, and K. Ohta. Attribute-based encryption with partially hidden encryptor-specified access structures. In S. M. Bellare, R. Gennaro, A. D. Keromytis, and M. Yung, editors, *ACNS 08*, volume 5037 of *LNCS*, pages 111–129, New York, NY, USA, June 3–6, 2008. Springer, Heidelberg, Germany.
- [39] T. Okamoto and K. Takashima. Hierarchical predicate encryption for inner-products. In M. Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 214–231, Tokyo, Japan, Dec. 6–10, 2009. Springer, Heidelberg, Germany.
- [40] T. Okamoto and K. Takashima. Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption. In D. Lin, G. Tsudik, and X. Wang, editors, *CANS 11*, volume 7092 of *LNCS*, pages 138–159, Sanya, China, Dec. 10–12, 2011. Springer, Heidelberg, Germany.
- [41] B. Parno, M. Raykova, and V. Vaikuntanathan. How to delegate and verify in public: Verifiable computation from attribute-based encryption. In R. Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 422–439, Taormina, Sicily, Italy, Mar. 19–21, 2012. Springer, Heidelberg, Germany.
- [42] C.-P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.
- [43] B. Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 53–70, Taormina, Italy, Mar. 6–9, 2011. Springer, Heidelberg, Germany.
- [44] R. Wei and D. Ye. Delegate predicate encryption and its application to anonymous authentication. In W. Li, W. Susilo, U. K. Tupakula, R. Safavi-Naini, and V. Varadharajan, editors, *ASIACCS 09*, pages 372–375, Sydney, Australia, Mar. 10–12, 2009. ACM Press.
- [45] S. Yamada, N. Attrapadung, B. Santoso, J. C. N. Schuldt, G. Hanaoka, and N. Kunihiro. Verifiable predicate encryption and applications to CCA security and anonymous predicate authentication. In M. Fischlin, J. Buchmann, and M. Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 243–261, Darmstadt, Germany, May 21–23, 2012. Springer, Heidelberg, Germany.

## A Security Analysis

In the following we analyze the security of our construction as defined in Section 3, by proving that it achieves all of the properties specified in Section 2.

**Lemma 1.** [Unforgeability] *Let* (Setup, KGen, Enc, Dec) *be an attribute-hiding predicate encryption scheme, (SKGen, Sig, Vf) be an existentially unforgeable digital signature scheme, and* ( $\mathcal{P}$ ,  $\mathcal{V}$ ) *be an equivocal and extractable commitment scheme, then FC is an unforgeable functional credential scheme.*

*Proof.* We use a series of games to gradually modify the experiment for unforgeability and prove that the success

probability of any PPT adversary  $\mathcal{A}$  is negligible in the security parameter.

**Game<sub>0</sub>.** Resembles exactly the experiment described in Definition 3.

**Game<sub>1</sub>.** The algorithm  $\text{DelShowCred}(\text{opk}, \text{tok})$  is modified to always abort the execution in step 4. if  $\text{tok}$  is not the output of a query of  $\mathcal{A}$  to the oracle  $\mathcal{O}_{\text{osk}}^{\text{Del}}$ .

**Game<sub>2</sub>.** The algorithm  $\text{CKGen}(1^\lambda)$  is modified by having  $\text{crs}$  generated by  $\mathcal{E}(1^\lambda)$  (along with  $\alpha$  that is kept locally).

**Game<sub>3</sub>.** In the next step, we modify the two algorithms  $\text{ShowCred}(\text{opk}, \text{cred}, f)$  and  $\text{DelShowCred}(\text{opk}, \text{cred}, \text{tok})$  as follows: in step 2.  $\text{com}_0$  is extracted by  $m' \leftarrow \mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}_0)$  and in step 6. the string  $s' || t'$  is computed as  $m' \oplus r'$ .

**Game<sub>4</sub>.** In this transition, we change the two algorithms  $\text{ShowCred}(\text{opk}, \text{cred}, f)$  and  $\text{DelShowCred}(\text{opk}, \text{cred}, \text{tok})$  as follows: in step 4.  $\text{com}_1$  is computed as  $\text{com}_1 \leftarrow \mathcal{E}_{\text{Eq}}^0(\text{crs}, \alpha)$ . In step 6. the algorithms compute  $\text{decom}_1 \leftarrow \mathcal{E}_{\text{Eq}}^1(\text{crs}, \alpha, \text{com}_1, m)$ .

**Game<sub>5</sub>.** In the next step, we modify the two algorithms  $\text{ShowCred}(\text{opk}, \text{cred}, f)$  and  $\text{DelShowCred}(\text{opk}, \text{cred}, \text{tok})$  as follows: in step 4 the decryption algorithm is no longer evaluated and in step 6. the algorithms compute  $\text{decom}_1 \leftarrow \mathcal{E}_{\text{Eq}}^1(\text{crs}, \alpha, \text{com}_1, s')$  if the attributes  $A$  associated to the input credential  $\text{cred}$  satisfy the input policy  $f$  (the policy  $f_{\text{tok}}$  associated with  $\text{tok}$  in case of delegated verification), or return  $\perp$  otherwise.

**Game<sub>6</sub>.** The algorithms  $\text{VrfyCred}(\text{opk}, f)$  and  $\text{DelVrfyCred}(\text{opk}, \text{tok})$  are changed as follows: in step 1. the commitment  $\text{com}_0$  is generated by  $\mathcal{E}_{\text{Eq}}^0(\text{crs}, \alpha)$  and in step 5.  $\text{decom}_0$  is generated by  $\mathcal{E}_{\text{Eq}}^1(\text{crs}, \alpha, \text{com}_0, r)$ .

**Game<sub>7</sub>.** The algorithms  $\text{VrfyCred}(\text{opk}, f)$  and  $\text{DelVrfyCred}(\text{opk}, \text{tok})$  are further changed as follows: in step 5. it is executed  $m' \leftarrow \mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}_1)$  and in step 7. the algorithm verifies whether  $s = m'$  and returns 1 if this is the case and 0 otherwise.

**Game<sub>0</sub>  $\approx$  Game<sub>1</sub>** Intuitively, the two games are indistinguishable because the probability of the adversary to produce a valid verification token without asking the corresponding oracle  $\mathcal{O}_{\text{osk}}^{\text{Del}}$  corresponds to the probability of forging a valid signature. Assume towards contradiction that there exists an adversary  $\mathcal{A}$  such that

$$|\Pr[1 \leftarrow \text{Game}_0^{\mathcal{A}}] - \Pr[1 \leftarrow \text{Game}_1^{\mathcal{A}}]| \geq \epsilon(\lambda)$$

for some non-negligible function  $\epsilon(\lambda)$ . Then we can build the following reduction against the existential unforge-

ability of the digital signature scheme  $(\text{SKGen}, \text{Sig}, \text{Vf})$ . The reduction is elaborated below.

$\mathcal{R}(1^\lambda, \text{vk})^{\mathcal{O}^{\text{Sig}}}$ : the reduction takes as input the security parameter  $1^\lambda$  and the verification key  $\text{vk}$  and has access to the oracle  $\mathcal{O}^{\text{Sig}}$ . It simulates the game that  $\mathcal{A}$  is expecting by inserting  $\text{vk}$  in the public key  $\text{opk}$  while the other elements are freshly generated with the appropriate algorithm. The oracles are simulated as dictated by the security definition except for  $\mathcal{O}_{\text{osk}}^{\text{Del}}$  where  $\mathcal{R}$  queries  $\mathcal{O}^{\text{Sig}}$  on  $c$  instead of signing it itself. Upon each query of  $\mathcal{A}$  to the oracle  $\mathcal{O}^{\text{DelShowCred}}$  on some  $\text{tok}_i$  it parses it as  $(c_i, \sigma_i)$ : if  $\text{Vf}(\text{vk}, c_i, \sigma_i) = 1$  and  $c_i$  was not queried to  $\mathcal{O}^{\text{Sig}}$ , then  $\mathcal{R}$  outputs  $(c_i, \sigma_i)$  and interrupts the simulation.

We note that the reduction is clearly efficient and faithfully simulates all the inputs that the adversary is expecting, except that the simulation is interrupted whenever the adversary guesses a valid fresh pair  $(c_i, \sigma_i)$ . We call such an event *forge*. By initial assumption we have that

$$|\Pr[1 \leftarrow \text{Game}_0^{\mathcal{A}}] - \Pr[1 \leftarrow \text{Game}_1^{\mathcal{A}}]| = \text{forge} \geq \epsilon(\lambda)$$

This represents a contradiction with respect to existential unforgeability of  $(\text{SKGen}, \text{Sig}, \text{Vf})$  and proves our claim.

**Game<sub>1</sub>  $\approx$  Game<sub>2</sub>** The difference between the two games lies only in the common reference string  $\text{crs}$ , thus any adversary that has a different success probability constitutes a valid distinguisher between a  $\text{crs}$  sampled uniformly at random and a trapdoor  $\text{crs}$ . More formally, assume that there exists an adversary that

$$|\Pr[1 \leftarrow \text{Game}_1^{\mathcal{A}}] - \Pr[1 \leftarrow \text{Game}_2^{\mathcal{A}}]| \geq \epsilon(\lambda)$$

for some non-negligible function  $\epsilon(\lambda)$ . We can construct the following reduction against the equivocability and extractability of  $(\mathcal{P}, \mathcal{V})$ .

$\mathcal{R}(\text{crs})$ : the reduction simulates **Game<sub>1</sub>** by plugging  $\text{crs}$  in the public parameters  $\text{opk}$ . If the adversary succeeds in the game it outputs 1, otherwise it returns 0.

It is easy to see that the reduction is efficient. Furthermore, whenever  $\text{crs}$  is sampled uniformly in  $\{0, 1\}^{\text{poly}(\lambda)}$ , then  $\mathcal{R}$  perfectly simulates **Game<sub>1</sub>**, on the other hand whenever  $\text{crs} \leftarrow \mathcal{E}(1^\lambda)$  then  $\mathcal{R}$  perfectly simulates **Game<sub>2</sub>**. Therefore we have that

$$\Pr[1 \leftarrow \text{Game}_1^{\mathcal{A}}] = \Pr[1 \leftarrow \mathcal{R} \mid \text{crs} \leftarrow \{0, 1\}^{\text{poly}(\lambda)}]$$

and

$$\Pr[1 \leftarrow \text{Game}_2^{\mathcal{A}}] = \Pr[1 \leftarrow \mathcal{R} \mid (\text{crs}, \alpha) \leftarrow \mathcal{E}(1^\lambda)].$$

It follows that

$$\left| \frac{\Pr[1 \leftarrow \mathcal{R} \mid \text{crs} \leftarrow \{0, 1\}^{\text{poly}(\lambda)}]}{\Pr[1 \leftarrow \mathcal{R} \mid (\text{crs}, \alpha) \leftarrow \mathcal{E}(1^\lambda)]} - 1 \right| \geq \epsilon(\lambda)$$

which represents a contradiction to the equivocability and extractability of the non-interactive commitment scheme. This proves our claim.

Game<sub>2</sub> ≈ Game<sub>3</sub> The two games are indistinguishable due to the fact that the probability of  $\mathcal{E}_{\text{Ext}}$  to extract the wrong message out of a commitment is negligible. We shall note that the only difference between the two games is that, for all executions of the ShowCred and DelShowCred algorithms, in Game<sub>2</sub> the strings  $s' || t'$  are computed as  $\mathcal{V}(\text{crs}, \text{com}_0, \text{decom}_0) \oplus r'$ , while in Game<sub>3</sub>  $s' || t'$  are computed as  $\mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}_0) \oplus r'$ . Thus the two games differ only whenever  $\mathcal{V}(\text{crs}, \text{com}_0, \text{decom}_0) \neq \mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}_0)$ . By the equivocability and extractability property of  $(\mathcal{P}, \mathcal{V})$  we have that

$$\Pr[\mathcal{V}(\text{crs}, \text{com}_0, \text{decom}_0) \neq \mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}_0)] \leq \text{negl}(\lambda)$$

therefore

$$|\Pr[1 \leftarrow \text{Game}_2^{\mathcal{A}}] - \Pr[1 \leftarrow \text{Game}_3^{\mathcal{A}}]| \leq \text{negl}(\lambda).$$

Game<sub>3</sub> ≈ Game<sub>4</sub> The indistinguishability of the two games follows again from the equivocability and extractability of the commitment scheme. In particular, assume towards contradiction that there exists an adversary  $\mathcal{A}$  such that

$$|\Pr[1 \leftarrow \text{Game}_3^{\mathcal{A}}] - \Pr[1 \leftarrow \text{Game}_4^{\mathcal{A}}]| \geq \epsilon(\lambda)$$

for some non-negligible function  $\epsilon(\lambda)$ . Then we can build the following distinguisher against the equivocability and extractability property of  $(\mathcal{P}, \mathcal{V})$ . The reduction looks as follows.

$\mathcal{R}(\text{crs})$ : the reduction simulates the inputs of Game<sub>3</sub> plugging  $\text{crs}$  in the public parameters  $\text{opk}$ . In the simulation of the oracles  $\mathcal{O}^{\text{ShowCred}}$  and  $\mathcal{O}^{\text{DelShowCred}}$  it modifies the algorithms ShowCred and DelShowCred as follows: in step 4 it evaluates the decryption and sends  $m \leftarrow \text{Dec}(\text{cred}, c)$  and sends  $m$  to the challenger. In response it receives the commitment  $\text{com}_1$ , then it proceeds with the execution until step 6. where it recomputes  $c$ . If the check succeeds, then  $\mathcal{R}$  sends  $m$  to the challenger and it receives  $\text{decom}_1$ , which is forwarded to  $\mathcal{A}$ .  $\mathcal{R}$  continues with the simulation and outputs 1 if the adversary succeeds and 0 otherwise.

It is easy to see that the reduction is efficient. We note that whenever  $\text{com}_1$  and  $\text{decom}_1$  are honestly computed, the reduction perfectly simulates the inputs that

the adversary is expecting Game<sub>3</sub>, while when the algorithms  $\mathcal{E}_{\text{Eq}}^0$  and  $\mathcal{E}_{\text{Eq}}^1$  are executed by the challenger,  $\mathcal{R}$  faithfully simulates Game<sub>4</sub>. It follows that

$$\Pr[1 \leftarrow \text{Game}_3^{\mathcal{A}}] = \Pr[1 \leftarrow \mathcal{R} \mid (\text{com}_1, \text{decom}_1) \leftarrow \mathcal{P}(\text{crs}, m)]$$

and

$$\Pr[1 \leftarrow \text{Game}_4^{\mathcal{A}}] = \Pr \left[ 1 \leftarrow \mathcal{R} \left[ \begin{array}{l} \text{com}_1 \leftarrow \mathcal{E}_{\text{Eq}}^0(\text{crs}, \alpha); \\ \text{decom}_1 \leftarrow \mathcal{E}_{\text{Eq}}^1(\text{crs}, \alpha, \text{com}_1, m) \end{array} \right] \right].$$

Therefore, by the initial assumption, we can estimate the success probability of the distinguisher as

$$\left| \frac{\Pr[1 \leftarrow \mathcal{R} \mid (\text{com}_1, \text{decom}_1) \leftarrow \mathcal{P}(\text{crs}, m)]}{\Pr \left[ 1 \leftarrow \mathcal{R} \left[ \begin{array}{l} \text{com}_1 \leftarrow \mathcal{E}_{\text{Eq}}^0(\text{crs}, \alpha); \\ \text{decom}_1 \leftarrow \mathcal{E}_{\text{Eq}}^1(\text{crs}, \alpha, \text{com}_1, m) \end{array} \right] \right]} - 1 \right| \geq \epsilon(\lambda).$$

Since  $\text{crs} \leftarrow \{0, 1\}^{\text{poly}(\lambda)} \approx \text{crs} \leftarrow \mathcal{E}(1^\lambda)$ , we have

$$\left| \frac{\Pr \left[ 1 \leftarrow \mathcal{R} \left[ \begin{array}{l} \text{crs} \leftarrow \{0, 1\}^{\text{poly}(\lambda)}; \\ (\text{com}_1, \text{decom}_1) \leftarrow \mathcal{P}(\text{crs}, m) \end{array} \right] \right]}{\Pr \left[ 1 \leftarrow \mathcal{R} \left[ \begin{array}{l} \text{crs} \leftarrow \mathcal{E}(1^\lambda); \\ \text{com}_1 \leftarrow \mathcal{E}_{\text{Eq}}^0(\text{crs}, \alpha); \\ \text{decom}_1 \leftarrow \mathcal{E}_{\text{Eq}}^1(\text{crs}, \alpha, \text{com}_1, m) \end{array} \right] \right]} - 1 \right| \gtrsim \epsilon(\lambda)$$

which is a contradiction to the equivocability and extractability property of  $(\mathcal{P}, \mathcal{V})$ . This proves our claim.

Game<sub>4</sub> ≈ Game<sub>5</sub> The indistinguishability of the two games follows from the correctness of the predicate encryption scheme and builds upon three main observations:

1. In the simulation of the algorithms ShowCred and DelShowCred, the policy  $f$  associated to the ciphertext  $c$  is always known to the challenger: in the former case it is given as in input, while in the latter it is associated with the input verification token  $\text{tok}$ , since all the verification tokens not coming from  $\mathcal{O}_{\text{osk}}^{\text{Del}}$  are discarded.
2. The ciphertext  $c$  is deterministically recomputed so it must be the case that it is generated by  $\text{Enc}(\text{pk}, f, \cdot)$ .
3. The ciphertext  $c$  is uniformly distributed on the ciphertext space: in order to see that we point out that the string  $s' || t'$  is computed as  $\mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}_0) \oplus r'$  and since  $r'$  is uniformly distributed and  $\mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}_0)$  is clearly independent from  $r'$ , then  $s' || t'$  is distributed uniformly in the set  $\{0, 1\}^{2\lambda}$ .

It follows that  $c$  is always a ciphertext of the form  $\text{Enc}(\text{pk}, f, \cdot)$ , for some well-defined policy  $f$ , uniformly distributed in its domain. Therefore by the correctness of the predicate encryption scheme (Setup, KGen, Enc, Dec) we have that, for all the set of attributes  $A$  associated with  $\text{cred}$  and for all  $m$ ,

- If  $f(A) = 1$  then  $\text{Dec}(\text{cred}, \text{Enc}(\text{pk}, A, m)) = m$ .
- If  $f(A) = 0$  then  $\text{Dec}(\text{cred}, \text{Enc}(\text{pk}, A, m)) = \perp$  with all but negligible probability.

Which implies that  $\text{Game}_5$  simulates  $\text{Game}_4$  with overwhelming probability.

$\text{Game}_5 \approx \text{Game}_6$  The two games are indistinguishable following from the equivocability and extractability of the commitment scheme. In particular, assume towards contradiction that there exists an adversary  $\mathcal{A}$  such that

$$|\Pr[1 \leftarrow \text{Game}_5^{\mathcal{A}}] - \Pr[1 \leftarrow \text{Game}_6^{\mathcal{A}}]| \geq \epsilon(\lambda)$$

for some non-negligible function  $\epsilon(\lambda)$ . Then we can build the following distinguisher against the equivocability and extractability property of  $(\mathcal{P}, \mathcal{V})$ . The reduction looks as follows.

$\mathcal{R}(\text{crs})$ : the reduction simulates the inputs of  $\text{Game}_5$  plugging  $\text{crs}$  in the public parameters  $\text{opk}$ . In the simulation of  $\text{VrfyCred}$  and  $\text{DelVrfyCred}$  it modifies the algorithms as follows: in step 1. the algorithms send  $r$  to the challenger and receive  $\text{com}_0$  in response. They proceed then with the normal execution until step 5. where they send  $r$  to the challenger and receive  $\text{decom}_0$ , which is forwarded to  $\mathcal{A}$ .  $\mathcal{R}$  continues with the simulation and outputs 1 if the adversary succeeds and 0 otherwise.

It is easy to see that the reduction is efficient. We note that whenever  $\text{com}_0$  and  $\text{decom}_0$  are honestly computed, the reduction perfectly simulates the inputs that the adversary is expecting  $\text{Game}_5$ , while when the algorithms  $\mathcal{E}_{\text{Eq}}^0$  and  $\mathcal{E}_{\text{Eq}}^1$  are executed by the challenger,  $\mathcal{R}$  faithfully simulates  $\text{Game}_6$ . It follows that

$$\Pr[1 \leftarrow \text{Game}_5^{\mathcal{A}}] = \Pr[1 \leftarrow \mathcal{R} \mid (\text{com}_0, \text{decom}_0) \leftarrow \mathcal{P}(\text{crs}, r)]$$

and

$$\Pr[1 \leftarrow \text{Game}_6^{\mathcal{A}}] = \Pr \left[ 1 \leftarrow \mathcal{R} \left| \begin{array}{l} \text{com}_0 \leftarrow \mathcal{E}_{\text{Eq}}^0(\text{crs}, \alpha); \\ \text{decom}_0 \leftarrow \mathcal{E}_{\text{Eq}}^1(\text{crs}, \alpha, \text{com}_0, r) \end{array} \right. \right].$$

Therefore, by the initial assumption, we can estimate the success probability of the distinguisher as

$$\left| \Pr[1 \leftarrow \mathcal{R} \mid (\text{com}_0, \text{decom}_0) \leftarrow \mathcal{P}(\text{crs}, r)] - \Pr \left[ 1 \leftarrow \mathcal{R} \left| \begin{array}{l} \text{com}_0 \leftarrow \mathcal{E}_{\text{Eq}}^0(\text{crs}, \alpha); \\ \text{decom}_0 \leftarrow \mathcal{E}_{\text{Eq}}^1(\text{crs}, \alpha, \text{com}_0, r) \end{array} \right. \right] \right| \geq \epsilon(\lambda)$$

which is a contradiction to the equivocability and extractability property of  $(\mathcal{P}, \mathcal{V})$ . This proves our claim.

$\text{Game}_6 \approx \text{Game}_7$  The two games are indistinguishable by the overwhelming probability of  $\mathcal{E}_{\text{Ext}}$  to extract the correct message out of a commitment generated by the adversary. It is clear that  $\text{Game}_6$  and  $\text{Game}_7$  only differ in the fact that the success of the adversary is determined by the check  $s = \mathcal{V}(\text{crs}, \text{com}_1, \text{decom}_1)$  in the former case and by  $s = \mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}_1)$ . Therefore the two games differ only whenever  $\mathcal{V}(\text{crs}, \text{com}_1, \text{decom}_1) \neq \mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}_1)$ . By the equivocability and extractability property of  $(\mathcal{P}, \mathcal{V})$  we have that

$$\Pr[\mathcal{V}(\text{crs}, \text{com}_1, \text{decom}_1) \neq \mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}_1)] \leq \text{negl}(\lambda)$$

therefore

$$|\Pr[1 \leftarrow \text{Game}_6^{\mathcal{A}}] - \Pr[1 \leftarrow \text{Game}_7^{\mathcal{A}}]| \leq \text{negl}(\lambda).$$

$\text{Game}_0 \approx \dots \approx \text{Game}_7$  By the previous propositions it follows that the success probability of the adversary is negligibly different between each pair of neighboring experiments. Since a polynomially-bounded sum of negligible functions is still a negligible function, we have that there is a negligible difference between the success probability of the adversary in  $\text{Game}_0$  and in  $\text{Game}_7$ . Therefore, in order to prove our lemma, it is enough to show that the advantage of the adversary in  $\text{Game}_7$  is bounded by a negligible function in the security parameter.

$\Pr[1 \leftarrow \text{Game}_7] \leq \text{negl}(\lambda)$  Intuitively, if the adversary would be able to win the game with more than negligible probability, then it would imply that the adversary is able to break the attribute-hiding property of the predicate encryption scheme with the same probability, which we assumed to be infeasible. More formally, assuming towards contradiction that there exists an adversary  $\mathcal{A}$  such that

$$|\Pr[1 \leftarrow \text{Game}_7]| \geq \epsilon(\lambda)$$

for some non-negligible function  $\epsilon(\lambda)$ . Then we can construct the following reduction against the attribute-hiding of the predicate encryption scheme

(Setup, KGen, Enc, Dec). The reduction is elaborated in the following.

$\mathcal{R}(\text{pk})$ : The reduction plugs  $\text{pk}$  into the public parameters  $\text{opk}$  and simulates  $\text{Game}_7$  by maintaining the same lookup lists with the only difference that the oracle  $\mathcal{O}_{\text{osk}}^{\text{GrantCred}}$ , when queried on a user id  $\text{id} \in \mathcal{H}$  simply records the corresponding set of attributes  $\mathbf{A}$  without generating the corresponding credential. The reduction instead generates credentials for the user id only upon a query id from  $\mathcal{A}$  to the oracle  $\mathcal{O}^{\text{CorrUser}}$ . Credentials are generated by querying the challenger on the Boolean formula for the corresponding set of attributes  $\mathbf{F}_{\mathbf{A}}$  and setting  $\text{cred}$  to be the answer  $\text{dk}_{\mathbf{F}_{\mathbf{A}}}$ . The rest of the oracles stay unchanged. At some point of the execution the adversary outputs either a policy  $f$  or a verification token  $\text{tok}$ : the challenger sets as  $f_{\text{tok}}$  the policy associated to the token  $\text{tok}$  and performs the same checks as dictated by the unforgeability experiment. In the simulation of  $\text{ShowCred}$  ( $\text{DelShowCred}$ , respectively),  $\mathcal{R}$  samples two random messages  $(r_0, r_1)$  and sends them to the challenger along with the tuple  $(f, f)$  ( $(f_{\text{tok}}, f_{\text{tok}})$ , respectively). The challenger returns  $c^*$  and  $\mathcal{R}$  sets  $c = c^*$  in step 3. of the algorithm. Once the adversary returns  $\text{com}_1$  in step 4.,  $\mathcal{R}$  runs  $m^* \leftarrow \mathcal{E}_{\text{Ext}}(\text{crs}, \alpha, \text{com}_1)$  and returns 1 if  $m^* = r_1$ , 0 if  $m^* = r_0$  otherwise it flips a random coin. The simulation of  $\mathcal{A}$  is interrupted at this point.

It is easy to see that the reduction runs in polynomial time. In the following we argue that the inputs the  $\mathcal{R}$  provides to the adversary perfectly resembles what  $\mathcal{A}$  is expecting from  $\text{Game}_7$ . We firstly point out that in the simulation of the oracles there is no noticeable difference: the decryption algorithm is in fact no longer evaluated in  $\mathcal{O}^{\text{ShowCred}}$  ( $\mathcal{O}^{\text{DelShowCred}}$ , respectively) and its execution depends only on the set of attributes  $\mathbf{A}$  associated with the input credential  $\text{cred}$ . Therefore it is sufficient to generate the credentials only when the adversary corrupts a certain user. In the last phase of the simulation we note that the uniform randomness  $r$  of the ciphertext  $c$  is revealed to  $\mathcal{A}$  only in step 5. (which is not executed due to the early interruption of  $\mathcal{A}$ ) and  $\text{com}_0$  is computed independently from  $r$ . Since the randomness of  $c^*$  is also sampled uniformly at random, it follows that the ciphertext  $c^*$  is correctly distributed according to the view of the adversary (up to step 4). Moreover it is easy to see that the tuples  $(r_0, r_1)$  and  $(f, f)$  ( $(f_{\text{tok}}, f_{\text{tok}})$ , respectively) are considered admissible by the challenger since the experiment requires that all the corrupted users (thus all the credential requested

to the challenger) do not satisfy the target policy  $f$  ( $f_{\text{tok}}$ , respectively). Now we provide a bound for the success probability of the reduction in the attribute-hiding game of the predicate encryption scheme. We shall remark that by assumption we have that  $|\Pr[1 \leftarrow \text{Game}_7]| \geq \epsilon(\lambda)$ , which means that with the same probability  $c^* = \text{Enc}(\text{pk}, \{f, f_{\text{tok}}\}, s) = \text{Enc}(\text{pk}, \{f, f_{\text{tok}}\}, m^*)$  (see the winning conditions of  $\text{Game}_7$ ). We denote this event by  $\text{guess}$ . Since the probability of the event  $\text{guess}$  to happen is independent from the random coins of the challenger (both  $r_0$  and  $r_1$  are correctly distributed) we can rewrite the success probability of the reduction in the attribute-hiding game as

$$\begin{aligned} \text{Adv}^{\mathcal{R}}(\lambda) &= \frac{1}{2} - \\ &\left( \frac{\Pr[1 \leftarrow \mathcal{R} \mid b = 1 \text{ and } \text{guess}]}{\Pr[0 \leftarrow \mathcal{R} \mid b = 0 \text{ and } \text{guess}]} \Pr[b = 1] + \right) \\ &\quad \cdot \text{guess} \\ &+ \left( \frac{\Pr[1 \leftarrow \mathcal{R} \mid b = 1 \text{ and } \overline{\text{guess}}]}{\Pr[0 \leftarrow \mathcal{R} \mid b = 0 \text{ and } \overline{\text{guess}}]} \Pr[b = 1] + \right) \\ &\quad \cdot \overline{\text{guess}}. \end{aligned}$$

Note that whenever the event  $\text{guess}$  does not happen we can upperbound the success probability of the reduction by  $\frac{1}{2}$ , therefore, by our initial assumption, we have that

$$\begin{aligned} \text{Adv}^{\mathcal{R}}(\lambda) &\gtrsim \frac{1}{2} - \\ &\left( \frac{\Pr[1 \leftarrow \mathcal{R} \mid b = 1 \text{ and } \text{guess}]}{\Pr[0 \leftarrow \mathcal{R} \mid b = 0 \text{ and } \text{guess}]} \Pr[b = 1] + \right) \epsilon(\lambda) \\ &+ \frac{1}{2} (1 - \epsilon(\lambda)). \end{aligned}$$

On the other hand whenever  $\text{guess}$  happens, the reduction successfully guesses the bit of the challenger with probability 1, therefore the advantage of  $\mathcal{R}$  is

$$\begin{aligned} \text{Adv}^{\mathcal{R}}(\lambda) &\gtrsim \frac{1}{2} - \left( 1 \cdot \epsilon(\lambda) + \frac{1}{2} \cdot (1 - \epsilon(\lambda)) \right) \\ &\gtrsim \frac{1}{2} - \left( \frac{1}{2} + \frac{1}{2} \epsilon(\lambda) \right) \\ &\gtrsim \frac{1}{2} \epsilon(\lambda) \end{aligned}$$

which is a non-negligible function in the security parameter. This is a contradiction to the attribute-hiding property of the predicate encryption scheme and it concludes our proof.  $\square$

**Lemma 2.** [Anonymity] *Let (Setup, KGen, Enc, Dec) be an attribute-hiding predicate encryption scheme, (SKGen, Sig, Vf) be an existentially unforgeable digital signature scheme, and  $(\mathcal{P}, \mathcal{V})$  be an equivocable and extractable commitment scheme, then FC is an anonymous functional credential scheme.*

*Proof.* In the proof we gradually modify the experiment through a series of games and then we argue about the success probability of the adversary in the last step.

Game<sub>0</sub> ≈ ⋯ ≈ Game<sub>5</sub> Here we define Game<sub>0</sub> as the experiment described in Definition 4 and we apply the same modifications as in the proof of unforgeability up to Game<sub>5</sub>. The proofs follow along the same lines.

$\Pr[1 \leftarrow \text{Game}_5] \leq \frac{1}{2} + \text{negl}(\lambda)$  The claim follows from a simple observation: in Game<sub>5</sub> the algorithms ShowCred and DelShowCred no longer evaluate the decryption using the credential cred<sub>b</sub> associated to the input  $i_b$ , rather it returns decom<sub>1</sub> depending on the bit  $f(A_b)$  ( $f_{\text{tok}}(A_b)$ , respectively), where  $A_b$  is the set of attributes associated with cred<sub>b</sub>. Since  $f_{\text{tok}}$  is always well defined (see Game<sub>1</sub>) and since the experiment requires that  $f(A_0) = f(A_1)$  ( $f_{\text{tok}}(A_0) = f_{\text{tok}}(A_1)$ , respectively), the random coin  $b$  of the oracles  $\mathcal{O}_b^{\text{Anon}}$  and  $\mathcal{O}_b^{\text{AnonDel}}$  is hidden in an information-theoretic sense. This implies that the advantage of the adversary is negligibly close to  $\frac{1}{2}$ , and it concludes our proof.  $\square$

**Lemma 3.** [Policy Hiding] *Let (Setup, KGen, Enc, Dec) be an attribute-hiding predicate encryption scheme, (SKGen, Sig, Vf) be an existentially unforgeable digital signature scheme, and (P, V) be an equivocable and extractable commitment scheme, then FC is a policy-hiding functional credential scheme.*

*Proof.* We prove the lemma through a series of games. The proof is elaborated below.

Game<sub>0</sub> ≈ ⋯ ≈ Game<sub>5</sub> Game<sub>0</sub> is defined as the experiment described in Definition 5 and we modify it as described in the proof of unforgeability up to Game<sub>5</sub>. It is easy to see that the indistinguishability argument still applies.

$\Pr[1 \leftarrow \text{Game}_5] \leq \frac{1}{2} + \text{negl}(\lambda)$  The proof of the claim follows from the attribute-hiding property of the predicate encryption scheme and we elaborate in the following why this is the case. Assume towards contradiction that there exists an adversary  $\mathcal{A}$  such that

$$\left| \Pr[1 \leftarrow \mathcal{A} \mid b = 1] - \Pr[1 \leftarrow \mathcal{A} \mid b = 0] \right| \geq \epsilon(\lambda)$$

for some non-negligible function  $\epsilon(\lambda)$ . Then we can construct the following reduction against the attribute-hiding of the predicate encryption scheme (Setup, KGen, Enc, Dec).

$\mathcal{R}(\text{pk})$ : The reduction punctures  $\text{pk}$  into the public parameters  $\text{opk}$  and faithfully simulates Game<sub>5</sub> except that

the credential for each user are initialized only when the user is corrupted by  $\mathcal{A}$ . For honest users  $\mathcal{R}$  simply records the attribute set  $A$  queried by the adversary and simulates the oracles  $\mathcal{O}^{\text{ShowCred}}$  and  $\mathcal{O}^{\text{DelShowCred}}$  as specified by the experiment. Once a user is corrupted,  $\mathcal{R}$  queries the challenger on all of the attribute sets  $A_i$  associated to it and sets each cred <sub>$i$</sub>  as the response  $\text{dk}_{F_A}$  of the challenger. The set of credentials is then handed over to the adversary. Upon a valid query  $(f_0, f_1)$  of the adversary to the oracle  $\mathcal{O}_b^{\text{PH}}$ ,  $\mathcal{R}$  forwards the tuple  $((1, 1), (f_0, f_1))$  to the challenger. The challenger returns in response a ciphertext  $c^*$  and  $\mathcal{R}$  sets  $\text{tok}_b = (c^*, \sigma = \text{Sig}(\text{sk}, c^*))$ . The reduction continues the simulation as specified in Game<sub>5</sub>. When  $\mathcal{A}$  outputs his guess  $b'$ ,  $\mathcal{R}$  forwards the bit to the challenger and interrupts the execution.

The reduction is clearly efficient. Furthermore it is easy to show that the simulation provided to  $\mathcal{A}$  perfectly mimics the inputs of Game<sub>5</sub>. Specifically, the modification to the oracles  $\mathcal{O}^{\text{ShowCred}}$  and  $\mathcal{O}^{\text{DelShowCred}}$  do not affect the view of the adversary since the execution of the functions ShowCred and DelShowCred (for honest users) does not depend on the credentials anymore, rather on the set of attributes associated to them (see Game<sub>5</sub>). Also  $\text{tok}_b$  is correctly distributed since  $c^*$  is a uniformly distributed encryption of 1 under either  $f_0$  or  $f_1$ . We shall point out that the tuple  $((1, 1), (f_0, f_1))$  sent to the challenger by the reduction is always admissible since for all the set of attributes  $A_i$  queried by the challenger it must hold that  $f_0(A) = f_1(A)$ . The same restrictions that the challenger applies to  $\mathcal{R}$  are then applied to the oracles offered to  $\mathcal{A}$ , therefore the simulation of  $\mathcal{R}$  is consistent throughout the whole experiment. What is left to be shown is that the success probability of  $\mathcal{A}$  carries over the advantage of  $\mathcal{R}$  in the attribute-hiding experiment. In order to see we note that whenever the random coin of the challenger is  $b = 1$ , then the reduction simulates the same choice Game<sub>5</sub> and the same holds for  $b = 0$ . Therefore it holds that  $\Pr[1 \leftarrow \mathcal{A} \mid b = 1] = \Pr[1 \leftarrow \mathcal{R} \mid b = 1]$  and that  $\Pr[1 \leftarrow \mathcal{A} \mid b = 0] = \Pr[1 \leftarrow \mathcal{R} \mid b = 0]$ . By the initial assumption we have that

$$\left| \Pr[1 \leftarrow \mathcal{R} \mid b = 1] - \Pr[1 \leftarrow \mathcal{R} \mid b = 0] \right| \geq \epsilon(\lambda)$$

which is a contradiction to the attribute-hiding property of the predicate encryption scheme. This concludes our proof.  $\square$