Noname manuscript No.
(will be inserted by the editor)

Attribute-Based Versions of Schnorr and ElGamal

Javier Herranz

Received: date / Accepted: date

Abstract We design in this paper the first attribute-based cryptosystems that work in the classical Discrete Logarithm, pairing-free, setting. The attribute-based signature scheme can be seen as an extension of Schnorr signatures, with adaptive security relying on the Discrete Logarithm Assumption, in the random oracle model. The attribute-based encryption schemes can be seen as extensions of ElGamal cryptosystem, with adaptive security relying on the Decisional Diffie-Hellman Assumption, in the standard model.

The proposed schemes are secure only in a bounded model: the systems admit L secret keys, at most, for a bound L that must be fixed in the setup of the systems. The efficiency of the cryptosystems, later, depends on this bound L. Although this is an important drawback that can limit the applicability of the proposed schemes in some real-life applications, it turns out that the bounded security of our key-policy attribute-based encryption scheme (in particular, with $L = 1$) is enough to implement the generic transformation of Parno, Raykova and Vaikuntanathan at TCC'2012. As a direct result, we obtain a protocol for the verifiable delegation of computation of boolean functions, which does not employ pairings or lattices, and whose adaptive security relies on the Decisional Diffie-Hellman Assumption.

Keywords Attribute-based cryptography · Discrete Logarithm setting · verifiable computation

Mathematics Subject Classification (2000) 94A60 · 94A62 · 68P25 · 15A03

Universitat Politècnica de Catalunya,
Dept. Matemàtica Aplicada IV
c. Jordi Girona 1-3, 08034, Barcelona, Spain
Tel.: +34-93-4016015
E-mail: jherranz@ma4.upc.edu

1 Introduction

Functional cryptography is emerging in the last years as a very interesting and powerful paradigm: decryptions or signatures can be now computed by several users, as long as they have enough rights, instead of by a unique user in possession of a secret key, as it is the case in the classical setting of public key cryptography. This more general setting of functional encryption seems much more suitable for real-life applications involving large amounts of different kinds of data, users and operations, such as storage and computation in the Cloud, big data analysis, social networks, or the Internet of Things.

Attribute-based cryptography is perhaps the particular case of functional encryption which has found more applications and, thus, has received more attention from the cryptographic community. In a typical attribute-based cryptosystem, the secret operation (signing or decrypting) can be performed only by users who hold a subset of attributes that satisfy some policy. Attribute-based cryptosystems must satisfy a collusion-resistance property: if a set of users, each of them holding attributes that do not satisfy the given policy, collude and try to perform the secret operation, they must fail to do so, even if the union of all their attributes satisfy the policy. This is the usual setting in attribute-based signature schemes [26, 28, 15, 4, 22] and in ciphertext-policy attribute-based encryption [7, 24, 25]. For encryption, the dual version of key-policy attribute-based encryption has also been defined and widely studied [20, 29, 9]: here the users' secret keys are related to policies, and ciphertexts are related to subsets of attributes; the ciphertext can be decrypted by a secret key only if the subset of attributes in the ciphertext is authorized for the policy in the secret key. Although the notion of ciphertext-policy may seem a bit more realistic, it turns out that key-policy attribute-based encryption has found some interesting applications, for instance in the area of verifiable delegation of computation [30].

The collusion-resistance property required to attribute-based cryptosystems makes it quite difficult to design secure systems. To do so, researchers have taken profit from the additional algebraic properties provided by mathematical objects like bilinear/multilinear maps or lattices. As a result, most of the attribute-based cryptosystems proposed up to now make use of lattices or multilinear maps; this includes very general constructions admitting arbitrary circuits as policies [18, 19, 8]. The only exceptions can be found in the area of attribute-based signatures, with [22] based on RSA, and some generic constructions [26, 4] that could in principle be implemented with RSA, as well. However, it is still desirable to study if attribute-based cryptography can be based on other classical techniques and security assumptions, like those in the traditional (pairing-free) Discrete Logarithm setting. The interest is both theoretical, to understand if more complex tools like pairings or lattices are necessary for building some cryptographic functionalities, and practical, because cryptosystems in the traditional Discrete Logarithm setting can be implemented in elliptic curves where elements in a group may have a shorter representation, which may lead to clear efficiency gains. We note that there

exist (at least) two recent examples where the same problem of building cryptographic protocols in the Discrete Logarithm pairing-free setting has been considered, in scenarios quite close to that of attribute-based cryptography. In [2], for the problem of anonymous credentials systems, which is very related to attribute-based signatures; and in [1], for the problem of inner-product encryption, which is another particular case of functional encryption.

1.1 Our Contributions

We propose in this work the first (to the best of our knowledge) attribute-based cryptosystems which use techniques and security assumptions of the pairing-free Discrete Logarithm setting. For attribute-based signatures, the new scheme is proved secure (private and unforgeable) in the random oracle model, under the only assumption that the Discrete Logarithm problem is hard to solve. For attribute-based encryption, we design both a ciphertext-policy scheme and a key-policy scheme. The security of both schemes is proved in the standard model, under the assumption that the Decisional Diffie-Hellman problem is hard. With these security properties in mind, and also due to the similarities in their designs, it is natural to consider the new schemes as the attribute-based versions of classical cryptosystems like Schnorr signatures [33] or ElGamal encryption [14].

A positive property of the schemes is that they achieve adaptive security, meaning that the schemes are secure even in front of adversaries that choose the challenge input (messages, policy and subset of attributes) in the challenge phase. This is in contrast to selective security, where the considered adversaries choose the challenge input in the setup phase of the systems. A negative property of the new schemes is the fact that they are *bounded*: a bound L on the maximum number of secret keys generated by the system must be chosen in the setup phase. Once the system has generated L keys, the setup phase must be run again, to generate new public parameters. Furthermore, the efficiency of the schemes (for instance the size of the public parameters, signatures and ciphertexts, and the computational cost of the protocols) depends on this bound L. This may be a serious limitation for possible uses of the schemes in some real-life applications, like social networks, with a huge expected number of users. Therefore, a direct implementation of the new schemes could make sense only in quite closed applications, for instance in small companies or institutions.

As we explain in the next Section 1.2, it seems hard to avoid this limitation when designing attribute-based cryptosystems in the pairing-free Discrete Logarithm setting. Interestingly, the bounded-security property may be enough in some applications of attribute-based cryptography. Indeed, a generic tranformation from a key-policy attribute-based encryption scheme to a protocol for the publicly verifiable delegation of computation is given in [30], where the security level required for the attribute-based primitive is *one-key security*. Actually, authors of [30] mention as an open problem the design of

one-key secure attribute-based cryptosystems with more efficiency or simplicity than the existing ones (which achieve unbounded security). Our results can be seen as an answer to this problem, because we show that nor pairings or lattices are needed in order to get one-key security. Combining our key-policy attribute-based encryption scheme with the construction in [30], one immediately obtains a protocol for the publicly verifiable delegation of computation of boolean functions, which does not require pairings or lattices, and whose adaptive security is based on the Decisional Diffie-Hellman Assumption.

1.2 Why to Bound the Number of Users?

In the attribute-based cryptosystems that we propose in this work, both the number (and name) of possible attributes and the number of users of the System must be bounded in advance. The first assumption is quite realistic and common in the attribute-based literature, since the possible attributes in real-life applications are usually known in advance, and not so many. However, imposing a bound on the number of users is a very strong limitation, which restricts the possible application of the proposed schemes to particular situations with few expected users, for instance providing attribute-based features to a small company or institution.

The question at this point is: can we avoid this drawback (bounding the number of users) in an attribute-based cryptosystem which does not use pairings, where the parameters are polynomial on the number of attributes, and where security relies only on the Discrete Logarithm family of assumptions (including the Discrete Logarithm and Computational / Decisional Diffie-Hellman ones)? We give now some informal arguments which seem to indicate that the answer to the previous question is 'No'. This will not be a formal (impossibility) result at all. In particular, our arguments are based on the current state-of-the-art in the area of digital signatures. Therefore, giving a final and formal answer to the previous question remains as an interesting and challenging open problem.

First of all, let us stress that we want to keep the parameters of the scheme polynomial on the (fixed) number N of attributes of the systems. If this is not a requirement, then it is quite easy to design attribute-based cryptosystems with the desired properties: if $\tilde{\mathcal{P}}$ denotes the global set of attributes, let us just generate one pair of ElGamal / Schnorr keys $(\mathsf{sk}_i, \mathsf{pk}_i)$ for each subset $A_i \subset \tilde{\mathcal{P}}$. The set of public keys will form the public parameters of the attribute-based system. When a user requests a secret key for a subset of attributes $S \subset \tilde{\mathcal{P}}$, he receives the secret keys sk_i for all subsets $S_i \subset S$. Note that the size of both the public parameters and secret keys are exponential on N. Later, for instance in the case of encryption, to encrypt a message for a decryption policy, one takes all the minimally authorized subsets $\{B_1, \ldots, B_k\}$ for that policy (a basis), and encrypts the plaintext for all the public keys associated to these sets (using multiple encryption techniques to re-use the randomness). If a user is authorized, some subset S_i in his secret key will match some of the subsets

B_j in the basis of the decryption policy, and so he could use the corresponding secret key sk_i to decrypt and obtain the plaintext. A similar solution for the case of (Schnorr) signatures is straightforward. It is easy to prove the selective security of the resulting cryptosystems.

Taking apart these solutions with exponential dependence on the number of attributes, the best we can do is to include, in the public parameters, some elements for each of the attributes in $\tilde{\mathcal{P}} = \{\mathsf{at}_1, \ldots, \mathsf{at}_N\}$. Since we want to work in the (pairing-free) Discrete Logarithm scenario, where we have a public group $\mathbb{G} = \langle g \rangle$ with prime order q, we can assume that we have some elements from $\mathbb{G}$ in the public parameters, whereas the master secret key contains the discrete logarithms (element in $\mathbb{Z}_q$) of those values. So let us assume that the public parameters contain $\{(g^{x_{i,1}}, \ldots, g^{x_{i,k}}\}_{\mathsf{at}_i \in \tilde{\mathcal{P}}}$, for some value k (polynomial in N), whereas the master secret key contains the corresponding discrete logarithms $\{(x_{i,1}, \ldots, x_{i,k}\}_{\mathsf{at}_i \in \tilde{\mathcal{P}}}$. When a user requests a secret key for a subset of attributes $S \subset \tilde{\mathcal{P}}$, he should get some secret, sensitive, unforgeable information related to the values $\{(x_{i,1}, \ldots, x_{i,k}\}_{\mathsf{at}_i \in S}$. Typically, as it happens in the identity-based setting, the values contained in the secret key can be thought as digital signatures computed by the master entity on the corresponding attributes; in the attribute-based setting, moreover, all the "signatures" must be linked, in order to prevent collusion attacks. We can distinguish three possibilities depending on the type of elements included in the resulting secret key sk_S:

(i) sk_S contains only elements from $\mathbb{G}$. In this case, even the verification of the validity of sk_S must involve the use of pairings, to check that the elements in sk_S and some elements from the public parameters satisfy some Diffie-Hellman relation. The same will happen later, in the signature/verification or encryption/decryption protocols. Actually, all the current pairing-based attribute-based cryptosystems belong to this category (i).

(ii) sk_S contains elements from $\mathbb{Z}_q$, and the verification of the validity of sk_S does involve the evaluation of some hash function. An example of this situation would be a system where each user can have at most one attribute at_i, and the secret key for that user would be a Schnorr signature (h_i, s_i) of message at_i, satisfying $h_i = H(g^{s_i} \cdot y^{-h_i}, \mathsf{at}_i)$, where y is part of the master public key. In the more realistic case where users may have more attributes, all the Schnorr signatures should be linked and randomized to prevent collusion attacks. This would lead to the appearance of (at least) one element R in the secret key sk_S, specific and different for each user, which furthermore must be added as an input of the hash evaluations, to preserve unforgeability of the secret keys. All in all, it seems impossible, for instance in the case of attribute-based signatures, that a user can later prove possession of some attributes without revealing some information on R or values h_i. This means that the resulting attribute-based signatures would be linkable, and so, the privacy requirement for attribute-based signatures would not be achieved. Note that this strategy is essentially the one proposed in [2] for an anonymous credential system based on DDH:

the credentials can be used at most once, because a second use of the same credentials would break anonymity. After the first use of the credentials, the user must go to the master entity to obtain new credentials.
Something similar happens in the case of encryption: if we do not want to rely on pairings, then it seems that the values R should be taken into account by the sender of the message when encrypting the plaintext, in order to allow later decryption. But this means that the sender must know in advance the identities of the users that will decrypt, and that those values R should be included in the public parameters. Both things are unrealistic.

(iii) sk_S contains elements from $\mathbb{Z}_q$, and the verification of the validity of sk_S does not involve the evaluation of any hash function. Taking into account that secret keys sk_S must be unforgeable, it seems we are now in the same situation as if we would like to design a secure digital signature scheme based on the Discrete Logarithm Assumption which does not use hash functions or pairings. The best that can be done in this situation, according to the current state-of-the-art, is to bound in advance (before defining the public key) the total number of messages that could be signed; otherwise, if the number of signatures available to an adversary is unlimited, he can infer enough equations between the signatures and the (unknown) elements of the secret key so that he gets the whole secret key and breaks the security of the system. In the digital signature setting, bounding the number of signatures led to the concept of k-times signature (with particular interest in the case $k = 1$) [32,6,27]. Translating this concept to our attribute-based setting, what we will get is a situation where the number of secret key queries is bounded. If we assume that the system can control that each user makes at most one secret key query, then what we get is a situation where the number of users participating in the system must be bounded in advance.

Summing up, our personal conclusion (after spending some time trying to design attribute-based cryptosystems in the pairing-free Discrete Logarithm setting) is a conjecture: in such a setting, and keeping all the parameters polynomial in N, the best one can do is to have systems where the number of users is bounded in advance. Once again, we encourage researchers to consider this problem and try to give a formal answer to the question discussed in this section.

1.3 Organization of the Paper

The rest of the paper is organized as follows. In Section 2 we describe the Discrete Logarithm setting, in particular the Discrete Logarithm and Decisional Diffie-Hellman Assumptions, and also we recall the definitions for zero-knowledge proofs of knowledge, which will be the main building block in the design of our attribute-based signature scheme.

Sections 3 and 4 are devoted to (bounded) attribute-based signatures: we first recall the syntax definition and the required security properties for such schemes, and then in Section 4 we describe the new scheme and prove its security, in the random oracle model, under the Discrete Logarithm Assumption. Both the design and the security analysis of the scheme are tightly related to a new zero-knowledge proof of knowledge for a language related to discrete logarithm relations. Section 5 contains the definitions of the protocols and the required security properties for (bounded) attribute-based encryption, in its two versions: ciphertext-policy and key-policy. We describe the new ciphertext-policy attribute-based encryption scheme in Section 6, where we prove its adaptive (but bounded) security in the standard model, under the Decisional Diffie-Hellman Assumption. Part of the security proof is related to an algebraic property of a (huge) matrix; the proof of this property, which involves simple but long linear algebra arguments, is moved to Appendix A in order to ease the global reading of the paper. Section 7 is the analogue of Section 6, but now for a new key-policy attribute-based encryption scheme.

For simplicity we describe and analyze our new schemes in the case of threshold policies; in Section 8 we explain how the schemes can be adapted in order to admit more general policies. We also discuss some (in)efficiency aspects of our schemes in that section, and we observe that our key-policy attribute-based encryption scheme can be combined with the construction in [30] to get a protocol for the publicly verifiable delegation and computation of boolean functions.

2 Mathematical Framework and Building-Blocks

The typical Discrete Logarithm framework consists of a cyclic group $\mathbb{G}$ of prime order q. Examples of such groups are subgroups of $\mathbb{Z}_p$, for some prime p, when $q|p-1$, or groups of points in an elliptic curve. Given a generator g of $\mathbb{G}$, and another element $y \in \mathbb{G}$, the discrete logarithm of y with respect to g is the integer $x \in \mathbb{Z}_q$ such that $g^x = y$.

We will assume the existence of some algorithm $(q, \mathbb{G}, g) \leftarrow \mathsf{DLog.Inst}(1^\lambda)$ which, on input a security parameter λ, outputs a triple $(q, \mathbb{G}, g)$, where q is a prime with λ bits, and $\mathbb{G} = \langle g \rangle$ is a cyclic group.

Definition 1 ***(Discrete Logarithm problem.)*** An algorithm $\mathcal{A}_{DLog}$ solves the Discrete Logarithm problem in $\mathbb{G}$ if it receives as input $(q, \mathbb{G}, g) \leftarrow \mathsf{DLog.Inst}(1^\lambda)$ and a randomly chosen $y \xleftarrow{R} \mathbb{G}$, and outputs $x \in \mathbb{Z}_q$ such that $g^x = y$.

The Discrete Logarithm Assumption states that the probability that any algorithm $\mathcal{A}_{DLog}$ solves the Discrete Logarithm problem in polynomial time is negligible in λ, meaning that this probability decreases (as λ increases) faster than the inverse of any polynomial.

Definition 2 ***(Decisional Diffie-Hellman problem.)*** An algorithm $\mathcal{A}_{DDH}$ solves the Decisional Diffie-Hellman (DDH, for short) problem in $\mathbb{G}$ if it receives as input $(q, \mathbb{G}, g) \leftarrow \mathsf{DLog.Inst}(1^\lambda)$ and a triple of elements (g^x, g^y, g^z)

in $\mathbb{G}$, where $x, y \xleftarrow{R} \mathbb{Z}_q$ are randomly chosen and z is either random or $z = xy \bmod q$, and is able to distinguish which is the case.

A bit more formally, the advantage of an algorithm $\mathcal{A}_{DDH}$ in solving the DDH problem is defined by considering the following experiment, $\mathbf{Exp}^{\mathsf{ddh}}_{\mathcal{A}_{DDH}}(\lambda)$, involving an adversary $\mathcal{B}$.

$\underline{\mathbf{Exp}^{\mathsf{ddh}}_{\mathcal{A}_{DDH}}(\lambda)}$
Choose $b \xleftarrow{R} \{0,1\}$ at random
$(q, \mathbb{G}, g) \leftarrow \mathsf{DLog.Inst}(1^\lambda)$
Choose $x, y \xleftarrow{R} \mathbb{Z}_q$ independently and at random
If $b = 0$, compute $T = g^{xy}$; if $b = 1$, sample $T \xleftarrow{R} \mathbb{G}$ independently and at random
$b' \leftarrow \mathcal{A}_{DDH}(q, \mathbb{G}, g, g^x, g^y, T)$
Output 1 if $b' = b$, and 0 otherwise.

The advantage of $\mathcal{A}_{DDH}$ in solving the DDH problem is defined as

$$\mathsf{Adv}^{\mathsf{ddh}}_{\mathcal{A}_{DDH}}(\lambda) = \left| \Pr[\mathbf{Exp}^{\mathsf{ddh}}_{\mathcal{A}_{DDH}}(\lambda) = 1] - \frac{1}{2} \right| .$$

The DDH Assumption states that the advantage of any polynomial time algorithm $\mathcal{A}_{DDH}$ in solving the DDH problem is negligible in λ.

2.1 Zero-Knowledge Proofs of Knowledge

Let $\mathcal{R}$ be a relation, containing pairs (x, w), such that, given (x, w), the fact $(x, w) \in \mathcal{R}$ can be verified in polynomial time. We will call x the statement and w the witness. We define the language $\mathcal{L}_{\mathcal{R}}$ as the set of statements x for which there exists a witness w such that $(x, w) \in \mathcal{R}$.

A zero-knowledge proof of knowledge (ZKPK) for a relation $\mathcal{R}$ is a (possibly interactive) protocol between two parties, a prover P and a verifier V, with common input x, where the prover convinces the verifier that he knows a witness w for which $(x, w) \in \mathcal{R}$, without revealing any additional information. Namely, the inputs for the prover are a statement x and a witness w such that $(x, w) \in \mathcal{R}$, whereas the input for the verifier is just x. At the end, the output for the verifier is 1 if it accepts the proof or 0 if it rejects it.

In this work we deal with a particular kind of ZKPK protocols, known as *Sigma* protocols, where the interaction consists of three steps. First of all, the prover computes and sends a commitment Cmt, then the verifier sends a challenge c, and finally the prover computes and sends an answer Ans. The final output of the verifier depends only on $\mathcal{R}, x, \mathsf{Cmt}, c, \mathsf{Ans}$. The *transcript* of an execution of this protocol is denoted as $(\mathsf{Cmt}, c, \mathsf{Ans})$.

For simplicity, we recall the security requirements of a ZKPK protocol for this particular case of Sigma protocols. A ZKPK must satisfy the following three properties:

2.1.1 Completeness.

Intuitively, this property ensures that, if the prover behaves honestly, then everything works fine and a valid proof is always accepted.

Definition 3 A ZKPK for a relation $\mathcal{R}$ is complete if, for all $(x, w) \in \mathcal{R}$, then an execution of the protocol where the input of the prover is (x, w) is always accepted by the verifier.

2.1.2 Knowledge Soundness.

Informally, this property guarantees that a (possibly malicious) prover who makes a proof be accepted as valid must actually know a witness. Let $W(x) = \{w \mid (x, w) \in \mathcal{R}\}$ denote the set of valid witnesses for element x.

Definition 4 A ZKPK for a relation $\mathcal{R}$ has knowledge soundness if there exists a polynomial-time extractor E such that, for any prover $\tilde{P}$ and any statement x, the probability that E, given access to transcripts of the protocol executed by $\tilde{P}$, outputs w such that $w \in W(x)$, is not significantly less than the probability that the executions run by $\tilde{P}$ are accepted as valid.

2.1.3 Zero-knowledge.

This property ensures that a valid proof for x does not reveal any information, other than the fact that there exists a witness w such that $(x, w) \in \mathcal{R}$. More formally,

Definition 5 A ZKPK for a relation $\mathcal{R}$ is zero-knowledge if, for any honest verifier $\tilde{V}$, there exists a polynomial-time simulator algorithm S such that for any $x \in \mathcal{L}_{\mathcal{R}}$, $\mathsf{S}(\mathcal{R}, x, \tilde{V})$ generates a transcript $(\mathsf{Cmt}, c, \mathsf{Ans})$ whose distribution is indistinguishable from the transcript of an execution of the protocol run by a honest prover, with input $(x, w) \in \mathcal{R}$, and verifier $\tilde{V}$.

This zero-knowledge property implies *witness indistinguishability* [16], which states that given a valid execution of the protocol for statement x, it is computationally hard to distinguish which witness in $W(x)$ was used by the prover. A stronger notion considers the possibility that verifier $\tilde{V}$ is malicious; however, since the role of the verifier in our signature scheme will be played by a hash function which behaves as a random function (and so, honestly), as sketched in the next paragraph, the zero-knowledge property with respect to honest verifiers will be enough.

The Fiat-Shamir [17] heuristics can be applied to a Sigma protocol in order to get a non-interactive zero-knowledge proof of knowledge protocol, where the whole elements in the proof are computed by the prover (the transcript is then usually denoted as π). The idea is to simply replace the challenge c computed by the verifier with the result of applying a hash function to the inputs $(x, \mathcal{R}, \mathsf{Cmt})$. In the random oracle model [5], where the hash function

is assumed to behave as a completely random function, this transformation preserves the security properties of the initial ZKPK protocol. If a message m is included as an additional input of the hash function, this technique allows the construction of signature schemes (known as *signatures of knowledge*), with security in the random oracle model. This is exactly what we will do in our signature scheme: a particular attribute-based signature will be a signature of knowledge, applied to the corresponding message, for a specific language that we will describe later.

3 (Bounded) Attribute-Based Signatures: Protocols and Security

In this section we describe the protocols that form an attribute-based signature scheme, as well as the security properties that must be required to such a scheme. An attribute-based signature is linked to a determined *signing policy* $(\mathcal{P}, \Gamma)$: a set $\mathcal{P}$ of attributes and a (monotone increasing) family $\Gamma \subset 2^{\mathcal{P}}$ of subsets of $\mathcal{P}$. A valid signature means that a signer possessing all the attributes of some of the subsets in Γ is the author of the signature. The monotonicity property ensures that $S_1 \subset S_2, S_1 \in \Gamma \ \Rightarrow S_2 \in \Gamma$. The most common and simple example of such a monotone increasing family of subsets is the threshold case: in a (t, n)-threshold signing policy, the set $\mathcal{P}$ contains n attributes, and $\Gamma = \{S \subset \mathcal{P} \ : \ |S| \geq t\}$. That is, by verifying a threshold attribute-based signature, the verifier is convinced that the author of the signature holds at least t of the attributes included in the set $\mathcal{P}$.

3.1 Syntactic Definition

A bounded attribute-based signature scheme consists of four probabilistic polynomial-time algorithms:

- $\mathsf{Setup}(1^\lambda)$. The setup algorithm takes as input a security parameter λ and outputs some public parameters pms and a master secret key msk. The public parameters contain the possible universe of attributes $\tilde{\mathcal{P}} = \{\mathsf{at}_1, \ldots, \mathsf{at}_N\}$ and a bound L for the maximum number of users.
- $\mathsf{KeyGen}(S, \mathsf{msk}, \mathsf{pms})$. The key generation algorithm takes as input the master secret key msk, the public parameters pms and then a set of attributes $S \subset \tilde{\mathcal{P}}$ satisfied by the user. The output is a secret key sk_S.
- $\mathsf{Sign}(m, \mathcal{P}, \Gamma, \mathsf{sk}_S, \mathsf{pms})$. The signing algorithm takes as input a message m, a signing policy $(\mathcal{P}, \Gamma)$ where $\mathcal{P} \subset \tilde{\mathcal{P}}$ and $\Gamma \subset 2^{\mathcal{P}}$, a secret key sk_S and the public parameters pms, and outputs a signature σ.
- $\mathsf{Verify}(\sigma, m, \mathcal{P}, \Gamma, \mathsf{pms})$. The verification algorithm takes as input the signature σ, the message m, the signing policy $(\mathcal{P}, \Gamma)$ and the public parameters pms, and outputs 1 (accept) or 0 (reject), depending on the validity of the signature.

Of course, the usual property of correctness must be required. Intuitively, a signature for a signing policy $(\mathcal{P}, \Gamma)$ that is computed by using sk_S such that $S \cap \mathcal{P} \in \Gamma$ must be always accepted by the verification protocol.

3.2 Security Definitions

Privacy.

Intuitively, privacy means that, given a valid signature, nobody (including the master entity and the stateless user who computed the signature) can obtain any information about the real author of the signature. In other words, given two subsets S_0 and S_1, with $S_0, S_1 \subset \mathcal{P}^*$, and a valid signature $\sigma \leftarrow \mathsf{Sign}(m, \mathcal{P}, \Gamma, \mathsf{sk}_{S_b}, \mathsf{pms})$ for a signing policy Γ such that $S_0, S_1 \in \Gamma$, nobody can guess the bit b with probability significantly bigger than $1/2$. The privacy property is formally defined via the following experiment $\mathbf{Exp}^{\mathsf{priv}}_{\mathcal{B}}(\lambda)$, involving an adversary $\mathcal{B}$.

$\underline{\mathbf{Exp}^{\mathsf{priv}}_{b,\mathcal{B}}(\lambda)}$
Choose $b \xleftarrow{R} \{0,1\}$ at random
$(\mathsf{pms}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$
$(m, \mathcal{P}, \Gamma, S_0, \mathsf{sk}_{S_0}, S_1, \mathsf{sk}_{S_1}, st_1) \leftarrow \mathcal{B}(\mathsf{pms}, \mathsf{msk})$
Verify that sk_{S_i} is a valid secret key for S_i, for $i = 0, 1$
Verify that $S_0 \cap \mathcal{P} \in \Gamma$ and $S_1 \cap \mathcal{P} \in \Gamma$
$\sigma^* \leftarrow \mathsf{Sign}(m, \mathcal{P}, \Gamma, \mathsf{sk}_{S_b}, \mathsf{pms})$
$b' \leftarrow \mathcal{B}(\sigma^*, \mathsf{pms}, \mathsf{msk}, st_1)$
Output 1 if $b' = b$, and 0 otherwise.

The advantage of $\mathcal{B}$ in breaking the privacy property is defined as

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathcal{B}}(\lambda) = \left| \Pr[\mathbf{Exp}^{\mathsf{priv}}_{\mathcal{B}}(\lambda) = 1] - \frac{1}{2} \right|.$$

Definition 6 An attribute-based signature scheme is private if, for any adversary $\mathcal{B}$ that runs in polynomial time, the advantage $\mathsf{Adv}^{\mathsf{priv}}_{\mathcal{B}}(\lambda)$ is negligible in the security parameter λ.

Since the adversary $\mathcal{B}$ can obtain the master secret key (and so, secret keys for all identities of his choice), it is easy to see that the privacy property, as defined above, implies other properties like signer's anonymity and non-linkability of different signatures.

Unforgeability.

An attribute-based signature scheme must satisfy the property of existential unforgeability against chosen message and signing policy attacks. Such property is defined by the following experiment $\mathbf{Exp}^{\mathsf{unf}}_{\mathcal{F}}(\lambda)$ involving an adversary $\mathcal{F}$.

$\underline{\mathbf{Exp}^{\mathsf{unf}}_{\mathcal{F}}(\lambda)}$
$(\mathsf{pms}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$
$(\sigma^*, m^*, \mathcal{P}^*, \Gamma^*) \leftarrow \mathcal{F}^{\mathsf{KeyGen}(\cdot,\mathsf{msk},\mathsf{pms}),\ \mathsf{Sign}(\cdot,\mathsf{pms})}(\mathsf{pms})$
Output 1 if all the following statements are true:
(i) $\mathsf{Verify}(\sigma^*, m^*, \mathcal{P}^*, \Gamma^*, \mathsf{pms})$ returns 1;
(ii) $\mathcal{F}$ has not made any secret key query S such that $S \cap \mathcal{P}^* \in \Gamma^*$;
(iii) the number of secret key queries is at most L (the bound given in pms);
(iv) $(m^*, \mathcal{P}^*, \Gamma^*, \sigma^*)$ is not the result of any signature query from $\mathcal{F}$.
Otherwise, output 0

The advantage of $\mathcal{F}$ in breaking the unforgeability of the scheme is defined as $\mathsf{Adv}^{\mathsf{unf}}_{\mathcal{F}}(\lambda) = \Pr[\mathbf{Exp}^{\mathsf{unf}}_{\mathcal{F}}(\lambda) = 1]$. We stress that $\mathcal{F}$ is allowed to make up to L adaptive queries for secret keys of subsets S of his choice, and adaptive signing queries for tuples $(m, \mathcal{P}, \Gamma)$ of his choice, where $\Gamma \subset 2^{\mathcal{P}}$. The last kind of queries are answered by choosing a random subset $S \subset \mathcal{P}$ with $S \in \Gamma$, and then by running $\mathsf{sk}_S \leftarrow \mathsf{KeyGen}(S, \mathsf{msk}, \mathsf{pms})$ and $\sigma \leftarrow \mathsf{Sign}(m, \mathcal{P}, \Gamma, \mathsf{sk}_S, \mathsf{pms})$.

Definition 7 An attribute-based signature scheme is unforgeable if, for any adversary $\mathcal{F}$ that runs in polynomial time, the advantage $\mathsf{Adv}^{\mathsf{unf}}_{\mathcal{F}}(\lambda)$ is negligible in the security parameter λ.

The above definition of unforgeability guarantees collusion resistance: a group of colluding users that pull their secret keys together will not be able to sign messages for a signing policy that none of the attribute sets of these users satisfies. The definition is in the *adaptive* setting where the attacker chooses the target signing policy $(\mathcal{P}^*, \Gamma^*)$ after making some queries. This is in contrast to the weaker *selective* setting where the attacker must choose the target signing policy at the very beginning of the attack.

4 The New Attribute-Based Signature Scheme (for Threshold Signing Policies)

In this section, we describe our attribute-based signature scheme. For simplicity, we consider the case of threshold signing policies. Therefore, a pair $(\mathcal{P}, \Gamma)$ will be represented as $(\mathcal{P}, t)$, where $1 \leq t \leq |\mathcal{P}|$. Later, we will explain how the scheme can be modified to support other (more general) signing policies.

4.1 The Proof of Knowledge

The language of the zero-knowledge proof of knowledge that will be used in the proposed attribute-based signature scheme is related to the public information $\mathbf{x} = \big(q, \mathbb{G}, g, h, \{Y_{i,j}\}_{1 \leq i \leq n, 1 \leq j \leq M}, \mathcal{P}, t\big)$, where The setup algorithm starts by running $(q, \mathbb{G}, g) \leftarrow \mathsf{DLog.Inst}(1^\lambda)$, which outputs a cyclic group $\mathbb{G} = \langle g \rangle$ is a

cyclig croup of prime order q, elements $h, \{Y_{i,j}\}_{1\leq i\leq n, 1\leq j\leq M}$ all belong to $\mathbb{G}$, and $(\mathcal{P}, t)$ is a signing policy.

A witness w for the fact that x belongs to this language consists of $\left(\mathbf{a}, S', \{s_i\}_{\mathsf{at}_i\in S'}\right)$, where $\mathbf{a} = (a_1, \ldots, a_M) \in (\mathbb{Z}_q)^M$, $S' \subset \mathcal{P}$ is a subset with $|S'| = t$ attributes and, for all $\mathsf{at}_i \in S'$, it holds that $s_i \in \mathbb{Z}_q - \{0\}$ and $g^{s_i} = \prod_{j=1}^{M} Y_{i,j}^{a_j}$. Without loss of generality, let us assume $\mathcal{P} = \{\mathsf{at}_1, \ldots, \mathsf{at}_n\}$ and $S' = \{\mathsf{at}_1, \ldots, \mathsf{at}_t\}$.

An interactive zero-knowledge Sigma protocol (three steps) where the prover proves the knowledge of a witness for such a statement can be constructed by combining existing techniques (see for instance [12,13,10]), as described below. We first provide the intuition of the protocol, and then we detail the steps of the protocol, in both its interactive and non-interactive versions.

4.1.1 A High-Level Description.

For simplicity, let us consider the case where the signing policy is simply $(\{\mathsf{at}_{i*}\}, 1)$, that is, there is only one attribute and the threshold is 1. Therefore, the proof of knowledge consists in proving knowledge of $\mathbf{a} = (a_1, \ldots, a_M) \in (\mathbb{Z}_q)^M$ and s_{i*} such that:

(i) $g^{s_{i*}} = \prod_{j=1}^{M} Y_{i*,j}^{a_j}$, and
(ii) $s_{i*} \in \mathbb{Z}_q - \{0\}$

This simple case can be solved by combining (via an "and" proof) the protocol in [10] to prove knowledge of relations between the exponents of public values, for the item (i), and the protocol in [12] to prove knowledge of an element s_{i*}^{-1} such that $(g^{s_{i*}})^{s_{i*}^{-1}} = 1$, for the item (ii).

However, the language that we consider here is more complex in general, because the signing policy can contain a set $\mathcal{P}$ with $n \geq 1$ attributes, the threshold can be any value $t \in \{1, 2, \ldots, n\}$, and the protocol must prove knowledge of at least t values $s_i \in \mathbb{Z}_q - \{0\}$, where $\mathsf{at}_i \in S'$ and $S' \subset \mathcal{P}$ is a subset with $|S'| = t$ attributes. The solution to go from the simpler case with a single attribute to this more complex case is to employ the techniques in [13]: therein, authors describe a method to transform a zero-knowledge proof of knowledge ZKPK_1 for a single element in a certain language into a zero-knowledge proof of knowledge $\mathrm{ZKPK}_{(t,n)}$ for a (t, n)-relation in that language: the second protocol proves knowledge of at least t witnesses for the belonging of t elements to the language, out of n public elements. The basic idea is to simulate executions of ZKPK_1 for $n - t$ instances for which the witness is unknown (thanks to the zero-knowledge property of ZKPK_1), to run real executions of ZKPK_1 for the t instances for which the witness is known, and to tie the n executions and the challenge with a polynomial with degree at most $n-t$; the polynomial is sent to the verifier, as well. The final verification step simply takes the n executions, checks their validity and checks that all of them are consistent with the polynomial.

This idea can be extended to the case of more general relations (or policies), not only threshold ones, by using the idea of dual access structures. The details can be found in [13].

4.1.2 The Interactive Version.

1. The prover generates the first message (commitment) of the Sigma protocol as follows:
   - For $j = 1, \ldots, M$ and for $i = 1, \ldots, n$, choose $r_i, \kappa_i, \delta_j \stackrel{R}{\leftarrow} \mathbb{Z}_q$, and compute $A_i = h^{r_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{a_j}$ and $T_i = h^{\kappa_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{\delta_j}$.
   - For $i = t+1, \ldots, n$, choose $c_i, u_i, \tilde{u}_i, z_i, \tilde{z}_i \stackrel{R}{\leftarrow} \mathbb{Z}_q$, and for each $i = t+1, \ldots, n$ and $j = 1, \ldots, M$, choose $w_{i,j} \stackrel{R}{\leftarrow} \mathbb{Z}_q$. Compute the values $U_i = A_i^{-c_i} \cdot g^{u_i} \cdot h^{z_i}$, $\tilde{A}_i = A_i^{\tilde{u}_i} \cdot h^{-\tilde{z}_i} \cdot g^{-c_i}$, $R_i = A_i^{-c_i} \cdot h^{z_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{w_{i,j}}$.
   - For $i = 1, \ldots, t$, choose $\alpha_i, \beta_i, \tilde{\alpha}_i, \tilde{\beta}_i \stackrel{R}{\leftarrow} \mathbb{Z}_q$, and for each $i = 1, \ldots, t$ and $j = 1, \ldots, M$, choose $\delta_{i,j} \stackrel{R}{\leftarrow} \mathbb{Z}_q$. Compute the values $U_i = g^{\alpha_i} \cdot h^{\beta_i}$, $\tilde{A}_i = A_i^{\tilde{\alpha}_i} \cdot h^{-\tilde{\beta}_i}$, $R_i = h^{\beta_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{\delta_{i,j}}$.

   The commitment sent by the prover is

$$\mathsf{Cmt} = \Big(\Big\{ \Big(A_i, \tilde{A}_i, T_i, R_i, U_i \Big) \Big\}_{1 \le i \le n} \Big).$$

2. The verifier chooses $c \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and sends the challenge c back to the prover.
3. Finally, the prover performs the following computations:
   - Find the (only) polynomial $f(x) \in \mathbb{Z}_q[X]$ with degree at most $n - t$ such that $f(0) = c \bmod q$ and $f(i) = c_i \bmod q$ for all $i = t+1, \ldots, n$.
   - For $i = 1, \ldots, t$, compute $c_i = f(i) \bmod q$ and then compute the values $u_i = \alpha_i + c_i s_i \bmod q$, $\tilde{u}_i = \tilde{\alpha}_i + c_i s_i^{-1} \bmod q$, $z_i = \beta_i + c_i r_i \bmod q$ and $\tilde{z}_i = \tilde{\beta}_i + c_i r_i s_i^{-1} \bmod q$.
   - For $i = 1, \ldots, t$ and for $j = 1, \ldots, M$, compute $w_{i,j} = \delta_{i,j} + c_i a_j \bmod q$.
   - For all $i = 1, \ldots, n$, compute the values $e_i = \kappa_i + c r_i \bmod q$. For all $j = 1, \ldots, M$, compute the values $w_j = \delta_j + c a_j \bmod q$.

   The final answer sent by the prover is

$$\mathsf{Ans} = \big(f(x), \{(u_i, \tilde{u}_i, z_i, \tilde{z}_i, e_i, \{w_{i,j}\}_{1 \le j \le M})\}_{1 \le i \le n} \ , \ \{w_j\}_{1 \le j \le M} \big).$$

In order to validate the correctness of the proof, the verifier outputs 1 if and only if the degree of $f(x)$ is at most $n-t$ and $f(0) = c$ and all the following equalities hold for all $i = 1, \ldots, n$, where $c_i = f(i) \bmod q$:

(1) $T_i = A_i^{-c} \cdot h^{e_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{w_j}$.

(2) $R_i = A_i^{-c_i} \cdot h^{z_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{w_{i,j}}$.

(3) $U_i = A_i^{-c_i} \cdot g^{u_i} \cdot h^{z_i}$.
(4) $\tilde{A}_i = A_i^{\tilde{u}_i} \cdot h^{-\tilde{z}_i} \cdot g^{-c_i}$.

4.1.3 The Non-Interactive Version.

We can apply the Fiat-Shamir heuristics to this Sigma protocol, by replacing the value $c \stackrel{R}{\leftarrow} Z_q$ chosen by the verifier with the output of a hash function $H : \{0,1\}^* \to \mathbb{Z}_q$ when computed by taking as inputs all the public values of the language and statement, $\mathbf{x}$, and also the values in Cmt. If we include as an additional input a message m to be signed, the resulting answer Ans is a signature of knowledge, on m, that proves that the author of the message knows a witness for the corresponding statement.

In this particular case, the signature on m would be

$$\sigma = \big(f(x), \{(A_i, u_i, \tilde{u}_i, z_i, \tilde{z}_i, e_i, \{w_{i,j}\}_{1\le j\le M})\}_{1\le i\le n} \ , \ \{w_j\}_{1\le j\le M}\big).$$

To verify the validity of such a signature for a message m and threshold policy $(\mathcal{P}, t)$, one checks that the degree of $f(x)$ is at most $n-t$, defines $c_i = f(i)$ for $i = 1, \ldots, n$, computes the values $T_i, R_i, U_i, \tilde{A}_i$ by following equations $(1), \ldots, (4)$ above, and finally checks if

$$f(0) = H\left(m, \mathbf{x}, \left\{\left(A_i, \tilde{A}_i, T_i, R_i, U_i\right)\right\}_{1\le i\le n}\right).$$

4.2 Security of the Proof of Knowledge

Let us prove that the interactive proof of knowledge that we have described in Section 4.1.2 achieves the necessary security level. Correctness of the zero-knowledge proof of knowledge is very easy to validate.

Regarding the zero-knowledge property, let us describe a suitable simulator S which works for any statement x in the language, and for any (possibly dishonest) verifier $\tilde{V}$. Recall that the goal of S is to produce a transcript $(\mathsf{Cmt}, c, \mathsf{Ans})$ whose distribution is indistinguishable from the transcript of an execution of the protocol run by a honest prover, with input $(x, w) \in \mathcal{R}$, and verifier $\tilde{V}$. The simulator S acts as follows:

1. Choose $c \stackrel{R}{\leftarrow} \mathbb{Z}_q$ with the same distribution as $\tilde{V}$ does.
2. Choose at random a polynomial $f(x) \in \mathbb{Z}_q[X]$ with degree at most $n-t$ such that $f(0) = c \bmod q$. Define $c_i = f(i) \bmod q$ for all $i = 1, \ldots, n$.
3. For all $i = 1, \ldots, n$, choose at random $A_i, \tilde{A}_i \stackrel{R}{\leftarrow} \mathbb{G}$ and $u_i, z_i, \tilde{u}_i, \tilde{z}_i, e_i \stackrel{R}{\leftarrow} \mathbb{Z}_q$. For all $j = 1, \ldots, M$, choose at random $w_j \stackrel{R}{\leftarrow} \mathbb{Z}_q$. Finally, for all $i = 1, \ldots, n$ and all $j = 1, \ldots, M$, choose at random $w_{i,j} \stackrel{R}{\leftarrow} \mathbb{Z}_q$.
4. For all $i = 1, \ldots, n$, compute the values $T_i = A_i^{-c} \cdot h^{e_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{w_j}$, $R_i = A_i^{-c_i} \cdot h^{z_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{w_{i,j}}$, $U_i = A_i^{-c_i} \cdot g^{u_i} \cdot h^{z_i}$, $\tilde{A}_i = A_i^{\tilde{u}_i} \cdot h^{-\tilde{z}_i} \cdot g^{-c_i}$.

5. Define the commitment as $\mathsf{Cmt} = \left(\left\{ \left(A_i, \tilde{A}_i, T_i, R_i, U_i \right) \right\}_{1 \leq i \leq n} \right)$, and the final answer as
   $\mathsf{Ans} = \left(f(x), \{(u_i, \tilde{u}_i, z_i, \tilde{z}_i, e_i, \{w_{i,j}\}_{1 \leq j \leq M})\}_{1 \leq i \leq n} \ , \ \{w_j\}_{1 \leq j \leq M} \right)$.
6. Output the transcript $(\mathsf{Cmt}, c, \mathsf{Ans})$.

It is easy to see that the distribution of this transcript is exactly the same as the distribution of a transcript generated by a honest prover P who knows a witness for $\mathbf{x}$, and $\tilde{V}$.

Finally, let us show that the proposed zero-knowledge proof of knowledge achieves the knowledge soundness property, assuming the hardness of the Discrete Logarithm problem.

Theorem 1 *Assuming the Discrete Logarithm problem is hard in $\mathbb{G}$, then the zero-knowledge proof of knowledge proposed in Section 4.1 achieves knowledge soundness.*

Proof Let us denote the success probability of $\tilde{P}$ as ϵ. The extractor E plays the role of the verifier and runs the zero-knowledge protocol with $\tilde{P}$ twice, in parallel, but for the same first message coming from $\tilde{P}$. As the second message of the protocol, E chooses two random but different challenges $c, c' \stackrel{R}{\leftarrow} \mathbb{Z}_q$, with $c \neq c'$. With probability ϵ^2, the two answers from $\tilde{P}$ to these two challenges are successful. If this is the case, E knows two accepted transcripts $(\mathsf{Cmt}, c, \mathsf{Ans})$ and $(\mathsf{Cmt}, c', \mathsf{Ans}')$. Let us denote them as

$$\mathsf{Cmt} = \left(\left\{ \left(A_i, \tilde{A}_i, T_i, R_i, U_i \right) \right\}_{1 \leq i \leq n} \right)$$

$$\mathsf{Ans} = \left(f(x), \{(u_i, \tilde{u}_i, z_i, \tilde{z}_i, e_i, \{w_{i,j}\}_{1 \leq j \leq M})\}_{1 \leq i \leq n} \ , \ \{w_j\}_{1 \leq j \leq M} \right)$$

$$\mathsf{Ans}' = \left(f'(x), \left\{ \left(u'_i, \tilde{u}'_i, z'_i, \tilde{z}'_i, e'_i, \{w'_{i,j}\}_{1 \leq j \leq M} \right) \right\}_{1 \leq i \leq n} \ , \ \{w'_j\}_{1 \leq j \leq M} \right).$$

Since the two transcripts are valid, we have $f(0) = c \neq c' = f'(0)$. The two polynomials have degree at most $n - t$, so we conclude that there must be at least t indices i such that $c_i = f(i) \neq f'(i) = c'_i$ (otherwise, the two polynomials would be equal in more than $n - t$ points and would therefore be the same polynomial, which contradicts the fact $f(0) \neq f'(0)$). Without loss of generality, let us assume that $c_i \neq c'_i$ for $i = 1, \ldots, t$.

The two answers satisfy equations $(1), \ldots, (4)$ in Section 4.1. Dividing the two valid instances of equation (1) for each index $i = 1, \ldots, n$, we have that $A_i^{c-c'} = h^{e_i - e'_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{w_j - w'_j}$ holds for all $i = 1, \ldots, n$. Let us denote $r_i = \frac{e_i - e'_i}{c - c'} \bmod q$ and $a_j = \frac{w_j - w'_j}{c - c'} \bmod q$, for all $j = 1, \ldots, M$, and thus we have

$$A_i = h^{r_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{a_j} \qquad \text{(Eq.1}i\text{)}$$

Dividing the two valid instances of equation (2) for each index $i = 1, \ldots, t$, we have that $A_i^{c_i - c_i'} = h^{z_i - z_i'} \cdot \prod_{j=1}^{M} Y_{i,j}^{w_{i,j} - w_{i,j}'}$ holds for all $i = 1, \ldots, t$. Denoting $\hat{r}_i = \frac{z_i - z_i'}{c_i - c_i'} \bmod q$ and $a_{i,j} = \frac{w_{i,j} - w_{i,j}'}{c_i - c_i'} \bmod q$, for all $j = 1, \ldots, M$ we have

$$A_i = h^{\hat{r}_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{a_{i,j}} \qquad \text{(Eq.2}i\text{)}$$

Now looking at equalities (Eq.1i) and (Eq.2i), for each index $i = 1, \ldots, t$, we have two representations of A_i with respect to the basis $h, Y_{i,1}, \ldots, Y_{i,M}$. Using a well-known result [11], under the assumption that the Discrete Logarithm problem is hard in $\mathbb{G}$, it turns out that the two representations must be the same. Therefore, we have $\hat{r}_i = r_i$ for all $i = 1, \ldots, t$, and we have $a_{i,j} = a_j$ for all $i = 1, \ldots, t$ and all $j = 1, \ldots, M$.

If we divide now the two valid instances of equation (3) for each index $i = 1, \ldots, t$, we have that $A_i^{c_i - c_i'} = g^{u_i - u_i'} \cdot h^{z_i - z_i'}$ holds for all $i = 1, \ldots, t$. Denoting $s_i = \frac{u_i - u_i'}{c_i - c_i'} \bmod q$, we have

$$A_i = g^{s_i} \cdot h^{\hat{r}_i} \qquad \text{(Eq.3}i\text{)}$$

Combining equalities (Eq.2i) and (Eq.3i), and using the fact that $r_i = \hat{r}_i$ for each index $i = 1, \ldots, t$ and $a_{i,j} = a_j$ for all $i = 1, \ldots, t$ and all $j = 1, \ldots, M$, we conclude that

$$g^{s_i} = \prod_{j=1}^{M} Y_{i,j}^{a_j},$$

for all $i = 1, \ldots, t$, as desired. Now we have to show that these values s_i are different from zero.

If we divide the two valid instances of equation (4) for each index $i = 1, \ldots, t$, we have that $g^{c_i - c_i'} = A^{\tilde{u}_i - \tilde{u}_i'} \cdot h^{\tilde{z}_i' - \tilde{z}_i}$ holds for all $i = 1, \ldots, t$. Let us note that $\tilde{u}_i - \tilde{u}_i' \neq 0$ under the assumption that Discrete Logarithm problem is hard in $\mathbb{G}$; otherwise, from the previous equality one gets the discrete logarithm of h with respect to the basis g. Denoting $\tilde{s}_i = \frac{\tilde{u}_i - \tilde{u}_i'}{c_i - c_i'} \bmod q$ and $\tilde{r}_i = \frac{\tilde{z}_i' - \tilde{z}_i}{c_i - c_i'} \bmod q$, we have $\tilde{s}_i \neq 0 \bmod q$ and

$$g = A_i^{\tilde{s}_i} \cdot h^{\tilde{r}_i} \qquad \text{(Eq.4}i\text{)}$$

Combining equalities (Eq.3i) and (Eq.4i), for each index $i = 1, \ldots, t$, we get $g^{1 - s_i \tilde{s}_i} = h^{\tilde{r}_i + \tilde{s}_i r_i}$. Under the assumption that the Discrete Logarithm is hard in $\mathbb{G}$, again, the only possibility is $1 - s_i \tilde{s}_i = 0 \bmod q$, which in particular means that $\tilde{s}_i = s_i^{-1} \bmod q$ and thus $s_i \neq 0 \bmod q$, for each $i = 1, \ldots, t$, as desired. □

4.3 Description of the Signature Scheme

$\mathsf{Setup}(1^\lambda)$. The setup algorithm starts by running $(q, \mathbb{G}, g) \leftarrow \mathsf{DLog.Inst}(1^\lambda)$, which outputs a cyclic group $\mathbb{G} = \langle g \rangle$ of prime order q, such that q is λ bits long. A cryptographic hash function $H : \{0,1\}^* \to \mathbb{Z}_q$ is chosen. The global set of attributes $\tilde{\mathcal{P}} = \{\mathsf{at}_1, \ldots, \mathsf{at}_N\}$ is chosen. A bound L for the maximum number of users in the system is given. Finally, the value $M = L+N$ is defined.

For all $i \in \{1, \ldots, N\}$ and all $j \in \{1, \ldots, M\}$, choose $x_{i,j} \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ independently and at random, and compute $Y_{i,j} = g^{x_{i,j}}$. An additional element $h \stackrel{R}{\leftarrow} \mathbb{G}$ is also randomly chosen.

The public parameters of the system are $\mathsf{pms} = (q, \mathbb{G}, g, h, H, \tilde{\mathcal{P}}, L, N, \{Y_{i,j}\}_{1 \le i \le N, 1 \le j \le M})$, whereas the master secret key is $\mathsf{msk} = \{x_{i,j}\}_{1 \le i \le N, 1 \le j \le M}$.

$\mathsf{KeyGen}(S, \mathsf{msk}, \mathsf{pms})$. The key generation algorithm takes as input a subset of attributes $S \subset \tilde{\mathcal{P}}$, the master secret key msk and the public parameters pms.

The master entity chooses at random a vector $\mathbf{a} = (a_1, \ldots, a_M) \stackrel{R}{\leftarrow} (\mathbb{Z}_q)^M$ and, for each $\mathsf{at}_i \in S$, computes the value $s_i = \sum_{j=1}^{M} a_j x_{i,j} \bmod q$. If some of the elements s_i is equal to zero (which happens with negligible probability), the master entity chooses a new vector $\mathbf{a}$.

The global secret key is $\mathsf{sk}_S = (\mathbf{a}, \{s_i\}_{\mathsf{at}_i \in S})$. In total, sk_S contains $M + |S|$ elements from $\mathbb{Z}_q$.

The receiver of the secret key can validate its correctness by checking that $g^{s_i} = \prod_{j=1}^{M} Y_{i,j}^{a_j}$, for all $\mathsf{at}_i \in S$.

$\mathsf{Sign}(m, \mathcal{P}, t, \mathsf{sk}_{\mathsf{id},S}, \mathsf{pms})$. The signing algorithm takes as input a message m, a set of attributes $\mathcal{P} \subset \tilde{\mathcal{P}}$, a threshold t, a secret key $\mathsf{sk}_S = (\mathbf{a}, \{s_i\}_{\mathsf{at}_i \in S})$ and the public parameters pms. The algorithm selects a minimally authorized set S', this is, a subset of $S \cap \mathcal{P}$ of cardinality exactly t. Without loss of generality and to simplify notation, let us assume $\mathcal{P} = \{\mathsf{at}_1, \ldots, \mathsf{at}_n\}$ and $S' = \{\mathsf{at}_1, \ldots, \mathsf{at}_t\}$. To generate the signature, the user runs the non-interactive zero-knowledge proof of knowledge protocol described in Section 4.1, with m as an additional input of the hash function (signature of knowledge), to compute

$$PK \left\{ \left(\mathbf{a}, S', \{s_i\}_{\mathsf{at}_i \in S'}\right) \text{ s.t. } S' \subset \mathcal{P} \bigwedge |S'| = t \bigwedge \forall \mathsf{at}_i \in S' : \left(s_i \neq 0 \ \wedge \ g^{s_i} = \prod_{j=1}^{M} Y_{i,j}^{a_j}\right) \right\} (m)$$

The resulting signature (following the notation in Section 4.1) is

$$\sigma = \left(f(x), \{(A_i, u_i, \tilde{u}_i, z_i, \tilde{z}_i, e_i, \{w_{i,j}\}_{1 \le j \le M})\}_{1 \le i \le n} \ , \ \{w_j\}_{1 \le j \le M}\right).$$

$\mathsf{Verify}(\sigma, m, \mathcal{P}, t, \mathsf{pms})$. The verification algorithm takes as input a message m, the signature σ on m, the threshold signing policy $(\mathcal{P}, t)$, with $n = |\mathcal{P}|$, and the public parameters pms. It simply checks the validity of the proof of knowledge. That is:

1. Verify that the degree of $f(x)$ is at most $n-t$.
2. For all $\mathsf{at}_i \in \mathcal{P}$, compute $c_i = f(i)$ and the values $T_i = A_i^{-c} \cdot h^{e_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{w_j}$, $R_i = A_i^{-c_i} \cdot h^{z_i} \cdot \prod_{j=1}^{M} Y_{i,j}^{w_{i,j}}$, $U_i = A_i^{-c_i} \cdot g^{u_i} \cdot h^{z_i}$, $\tilde{A}_i = A_i^{\tilde{u}_i} \cdot h^{-\tilde{z}_i} \cdot g^{-c_i}$.
3. Return 1 if $f(0) = H\left(m, \mathbf{x}, \left\{\left(A_i, \tilde{A}_i, T_i, R_i, U_i\right)\right\}_{\mathsf{at}_i \in \mathcal{P}}\right)$, and return 0 otherwise. Here we are denoting $\mathbf{x} = \left(g, h, \{Y_{i,j}\}_{1 \le i \le n, 1 \le j \le M}, \mathcal{P}, t\right)$.

Remark 1 If some vector $\mathbf{a}$ given to some user as part of his secret key was a linear combination of other vectors $\mathbf{a}^{(\ell)}$ given to other users, then the security of the scheme could be compromised, since the system could not resist a collusion attack. The probability of such a linear dependence is negligible in the security parameter λ. However, in a real implementation of this scheme, it could be desirable to explicitly check that such dependence relations do not occur. This can be done by the master entity, for instance, by choosing the L vectors $\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(L)}$, linearly independent in $(\mathbb{Z}_q)^M$, during the Setup protocol, and then assigning vector $\mathbf{a}^{(\ell)}$ to the ℓ-th secret key query (in a stateful process). See also Section 4.6 for a different and more efficient generation of the L vectors $\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(L)}$.

4.4 Privacy of the Signature Scheme

This property is achieved, in the random oracle model, because two valid signatures computed with two different secret keys for the same signing policy are actually two valid proofs of knowledge for the same statement, but computed with different witnesses. We have proved that the interactive proof of knowledge protocol underlying our attribute-based signature scheme achieves the (honest-verifier) zero-knowledge property in a perfect way. This implies [16] the perfect witness indistinguishability of that proof of knowledge, against honest verifiers. When the interactive protocol is turned into a signature scheme via the Fiat-Shamir heuristic, the witness indistinguishability property is preserved, in the random oracle model for the hash function H.

4.5 Unforgeability of the Signature Scheme

Theorem 2 *Assuming that the Discrete Logarithm problem is hard in $\mathbb{G}$, then the proposed attribute-based signature scheme satisfies the unforgeability property. The proof is in the random oracle model for H.*

Proof We are going to prove that, if there exists some adversary $\mathcal{F}$ which breaks the unforgeability property of the scheme with probability ε, then we

can solve the Discrete Logarithm problem with probability $\approx \frac{\varepsilon^2}{q_H^2}$, where q_H is the number of queries that $\mathcal{F}$ makes to the random oracle.

Let $(q, \mathbb{G}, g, y)$ be the instance of the Discrete Logarithm problem that we want to solve. We start running the experiment $\mathbf{Exp}^{\mathsf{unf}}_{\mathcal{F}}(\lambda)$ with adversary $\mathcal{F}$, by choosing the public parameters of the scheme. To do so, we choose the global set of attributes $\tilde{\mathcal{P}} = \{\mathsf{at}_1, \ldots, \mathsf{at}_N\}$ and the bound L for the maximum number of users in the system (and so, for the number of extraction queries that $\mathcal{F}$ can make). We define the value $M = L + N$. We choose $\tau \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and define $h = y^\tau$.

To define the values $\{Y_{i,j}\}_{1 \le i \le N, 1 \le j \le M}$, we choose N vectors $\boldsymbol{\theta}_i = (\theta_{i,1}, \ldots, \theta_{i,M}) \stackrel{R}{\leftarrow} (\mathbb{Z}_q)^M$, for $i = 1, \ldots, N$, randomly, in general position. With overwhelming probability, these N vectors will be linearly independent (we repeat the random choice if this is not the case). We choose N other vectors $\boldsymbol{\mu}_i = (\mu_{i,1}, \ldots, \mu_{i,M}) \stackrel{R}{\leftarrow} (\mathbb{Z}_q)^M$, for $i = 1, \ldots, N$, also randomly. For each $i = 1, \ldots, N$ and each $j = 1, \ldots, M$, we define $Y_{i,j} = g^{\mu_{i,j}} \cdot y^{\theta_{i,j}}$.

The public parameters of the system that we give to $\mathcal{F}$ are $\mathsf{pms} = (q, \mathbb{G}, g, h, H, \tilde{\mathcal{P}}, L, N, \{Y_{i,j}\}_{1 \le i \le N, 1 \le j \le M})$, where the description of H is simply "random oracle". Therefore, we maintain a table TAB_H where we introduce and store the input-ouput relations of this oracle. Each time $\mathcal{F}$ makes a hash query, we check if the input is already in TAB_H; if this is the case, we answer with the corresponding output; if the input is new, we choose an output randomly in $\mathbb{Z}_q$, we answer with this output, and we introduce a new entry in TAB_H, with this new input-output relation.

During the experiment, $\mathcal{F}$ will make up to L extraction queries, to obtain secret keys for subsets S of his choice, where $S \subset \tilde{\mathcal{P}}$, and also will make signing queries for messages m and (threshold) signing policies $(\mathcal{P}, t)$. We have to give correct answers to all such queries. To answer the ℓ-th extraction query, where $\ell \in \{1, \ldots, L\}$, corresponding to some subset of attributes $S^{(\ell)}$, we choose $\mathbf{a}^{(\ell)} = (a_1^{(\ell)}, \ldots, a_M^{(\ell)}) \stackrel{R}{\leftarrow} (\mathbb{Z}_q)^M$ at random such that $\mathbf{a}^{(\ell)} \cdot \boldsymbol{\theta}_i = 0 \bmod q$, for all $\mathsf{at}_i \in S^{(\ell)}$, and such that $\mathbf{a}^{(\ell)}$ is linearly independent with $\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(\ell-1)}$. For each $\mathsf{at}_i \in S^{(\ell)}$, we compute $s_i = \sum_{j=1}^{M} a_j^{(\ell)} \mu_{i,j}$. It is easy to check that the resulting secret key $\mathsf{sk}_{S^{(\ell)}} = (\mathbf{a}^{(\ell)}, \{s_i\}_{\mathsf{at}_i \in S^{(\ell)}})$ is valid.

Since $M = L + N$ and N is an upper bound for the number of attributes in such a subset S, it is always possible to find such L linearly independent vectors $\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(L)}$, and the secret keys obtained by $\mathcal{F}$ in these queries follow the same distribution as in the description of the scheme.

Regarding the signing queries for m and $(\mathcal{P}, t)$, we can answer them by following essentially the same procedure as the simulator S in the proof of the zero-knowledge property of the proposed proof of knowledge (in Section 4.2). The only difference is that, once the transcript $(\mathsf{Cmt}, c, \mathsf{Ans})$ has been generated, we have to add the relation $H(m, \mathbf{x}, \mathsf{Cmt}) = c$ to the hash table TAB_H (only with negligible probability this input for H had been queried before, as a hash query, by $\mathcal{F}$; we abort the experiment if this is the case). Here, $\mathbf{x}$ denotes the statement of the language, which contains the public

parameters and also the signing policy $(\mathcal{P}, t)$. The signature sent back to $\mathcal{F}$ is $\sigma = \mathsf{Ans}$.

With non-negligible probability ε, this forger $\mathcal{F}$ outputs a valid and non-trivial signature $(\sigma, m, \mathcal{P}, t)$, where $\mathcal{P} \subset \tilde{P}$ contains n attributes, $1 \le t \le n$ and

$$\sigma = \big(f(x), \{(A_i, u_i, \tilde{u}_i, z_i, \tilde{z}_i, e_i, \{w_{i,j}\}_{1\le j\le M})\}_{1\le i\le n}\ ,\ \{w_j\}_{1\le j\le M}\big)$$

satisfying in particular $c = f(0) = H(\mathsf{query})$, where $\mathsf{query} = \left(m, \mathbf{x}, \left\{\left(A_i, \tilde{A}_i, T_i, R_i, U_i\right)\right\}_{1\le i\le n}\right)$ and the values $T_i, R_i, U_i, \tilde{A}_i$ are computed with equations $(1), \ldots, (4)$ in Section 4.1.

Now the idea is to use the *replay technique*, also known as *forking lemma* [31]: the experiment $\mathbf{Exp}^{\mathsf{unf}}_{\mathcal{F}}(\lambda)$ is run again, with the same adversary $\mathcal{F}$ and using the same randomness and the same random oracle answers, until the query $H(\mathsf{query})$ corresponding to the first forged signature is made. At this time, a different value $c' \xleftarrow{R} \mathbb{Z}_{q'}, c' \neq c$ is chosen and returned as the answer to this hash query, and the experiment goes on. With some (non-negligible) probability $\approx \frac{\varepsilon}{q_H^2}$, the forger $\mathcal{F}$ will produce another valid and non-trivial forgery $(\sigma', m', \mathcal{P}', t')$, where

$$\sigma' = \big(f'(x), \{(A'_i, u'_i, \tilde{u}'_i, z'_i, \tilde{z}'_i, v'_i, e'_i, \{w'_{i,j}\}_{1\le j\le M})\}_{1\le i\le n}\ ,\ \{w'_j\}_{1\le j\le M}\big)$$

and $c' = f'(0) = H(\mathsf{query}')$, satisfying $\mathsf{query}' = \mathsf{query}$. This means that $\mathcal{P}' = \mathcal{P}$, $t' = t$, $m' = m$, $A'_i = A_i$, $\tilde{A}'_i = \tilde{A}_i$, $T'_i = T_i$, $R'_i = R_i$ and $U'_i = U_i$. On the other hand, since $f(0) = c \neq c' = f'(0)$ and the two polynomials have degree at most $n - t$, where $n = |\mathcal{P}|$, there must be a subset $\mathcal{I} \subset \mathcal{P}$ with $|\mathcal{I}| \ge t$ such that $f(i) = c_i \neq c'_i = f'(i)$, for all $\mathsf{at}_i \in \mathcal{I}$ (otherwise, the two polynomials would be the same one).

From this point, the analysis is exactly the same as in the proof of Theorem 1. From the two forgeries we can extract a vector $\mathbf{a} = (a_1, \ldots, a_M) \in (\mathbb{Z}_q)^M$ and values $\{s_i\}_{\mathsf{at}_i\in\mathcal{I}}$, where $s_i \in \mathbb{Z}_q^*$, such that

$$g^{s_i} = \prod_{j=1}^{M} Y_{i,j}^{a_j}, \quad \forall \mathsf{at}_i \in \mathcal{I}. \tag{Eq.6}$$

According to the relation between vector $\mathbf{a}$ and vectors $\{\mathbf{a}^{(\ell)}\}_{1\le\ell\le L}$ that we gave to $\mathcal{F}$ in the queried secret keys, we can distinguish two cases.

Case 1: a is linearly independent to $\{\mathbf{a}^{(\ell)}\}_{1\le\ell\le L}$.

Let us take an attribute $\mathsf{at}_i \in \mathcal{I}$. Note that the corresponding vector $\boldsymbol{\theta}_i = (\theta_{i,1}, \ldots, \theta_{i,M}) \in (\mathbb{Z}_q)^M$ is perfectly hidden in the public values $Y_{i,j} = g^{\mu_{i,j}} \cdot y^{\theta_{i,j}}$, for $j = 1, \ldots, M$. The only information that the attacker $\mathcal{F}$ could have obtained from $\boldsymbol{\theta}_i$ is derived from the extraction queries. Assuming that $\mathsf{at}_i \in S^{(\ell)}$ for all the extraction queries, what $\mathcal{F}$ knows is that $\boldsymbol{\theta}_i \cdot \mathbf{a}^{(\ell)} = 0$, for all $\ell = 1, \ldots, L$. But in this Case 1, we know that $\mathbf{a}$ is linearly independent to $\{\mathbf{a}^{(\ell)}\}_{1\le\ell\le L}$. Let us denote this subspace of $(\mathbb{Z}_q)^M$ as $G = \langle\{\mathbf{a}^{(\ell)}\}_{1\le\ell\le L}\rangle$. We

have $\dim G = L$ and $\dim(\mathsf{Ker}\{\boldsymbol{\theta}_i\}) = M - 1 > L$, and also we are assuming $G \subset \mathsf{Ker}\{\boldsymbol{\theta}_i\}$. The probability (over the random choices we have taken during the reduction) that $\mathbf{a} \in \mathsf{Ker}\{\boldsymbol{\theta}_i\}$ once we know that $\mathbf{a} \notin G$ is $1/q$. This can be seen, for instance, by extending the basis $\{\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(L)}\}$ of G to a basis of $\mathsf{Ker}\{\boldsymbol{\theta}_i\}$) (by adding $M - 1 - L$ vectors) and finally extending this basis to a basis of the whole space $(\mathbb{Z}_q)^M$ by adding a last vector. The expression of $\mathbf{a}$ with respect to this basis will have some element different to 0 in the last N positions, since $\mathbf{a} \notin G$; the rest of coefficients are randomly distributed. The probability that $\mathbf{a} \in \mathsf{Ker}\{\boldsymbol{\theta}_i\}$ is the probability that the last coefficient in the above-mentioned representation of $\mathbf{a}$ is equal to 0, which is $1/q$.

Therefore, with overwhelming probability we have $\mathbf{a} \notin \mathsf{Ker}\{\boldsymbol{\theta}_i\}$ in this case. From the equality (Eq.6) applied to this attribute at_i, we infer $g^{s_i} = g^{\boldsymbol{\mu}_i \cdot \mathbf{a}} \cdot y^{\boldsymbol{\theta}_i \cdot \mathbf{a}}$, with $\boldsymbol{\theta}_i \cdot \mathbf{a} \neq 0 \bmod q$. Therefore, we conclude that $x = \frac{s_i - \boldsymbol{\mu}_i \cdot \mathbf{a}}{\boldsymbol{\theta}_i \cdot \mathbf{a}} \bmod q$ satisfies $g^x = y$ and thus we solve the given instance of the Discrete Logarithm problem.

Case 2: a is a linear combination of vectors in $\{\mathbf{a}^{(\ell)}\}_{1 \leq \ell \leq L}$.

Let us write $\mathbf{a} = \sum_{1 \leq \ell \leq L} \psi_\ell \mathbf{a}^{(\ell)}$ and let us take some $\ell^* \in \{1, \ldots, L\}$ such that $\psi_{\ell^*} \neq 0 \bmod q$. Going back to the ℓ^*-th extraction query, for a subset of attributes $S^{(\ell^*)}$, by definition of a successful forgery, we know that the number of attributes in $S^{(\ell^*)} \cap \mathcal{P}$ is less than t. Therefore, there exists at least an attribute $\mathsf{at}_i \in \mathcal{I} - S^{(\ell^*)}$. Let us now define the subset of extraction queries $\mathcal{J}_1 = \{\ell \in \{1, \ldots, L\} \text{ s.t. } \mathsf{at}_i \in S^{(\ell)}\}$, and its complement $\mathcal{J}_2 = \{1, \ldots, L\} - \mathcal{J}_1$. We know that $\ell^* \in \mathcal{J}_2$ and that $\mathbf{a}^{(\ell)} \cdot \boldsymbol{\theta}_i = 0 \bmod q$, for all $\ell \in \mathcal{J}_1$. Let us write and denote

$$\mathbf{a} = \left(\sum_{\ell \in \mathcal{J}_1} \psi_\ell \mathbf{a}^{(\ell)} \right) + \left(\sum_{\ell \in \mathcal{J}_2} \psi_\ell \mathbf{a}^{(\ell)} \right) = \mathbf{a}_{\mathcal{J}_1} + \mathbf{a}_{\mathcal{J}_2},$$

where $\mathbf{a}_{\mathcal{J}_1} \cdot \boldsymbol{\theta}_i = 0 \bmod q$ and $\mathbf{a}_{\mathcal{J}_2} = \psi_{\ell^*} \mathbf{a}^{(\ell^*)} + \sum_{\ell \in \mathcal{J}_2, \ell \neq \ell^*} \psi_\ell \mathbf{a}^{(\ell)}$. Now $\mathbf{a} \in \mathsf{Ker}\{\boldsymbol{\theta}_i\}$ if and only if $\mathbf{a}_{\mathcal{J}_2} \in \mathsf{Ker}\{\boldsymbol{\theta}_i\}$. Furthermore, since all the vectors $\mathbf{a}^{(\ell)}$ are linearly independent, we know that $\mathbf{a}_{\mathcal{J}_2} \notin G = \langle \{\mathbf{a}^{(\ell)}\}_{\ell \in \mathcal{J}_1} \rangle$. Therefore, we are in the same situation as in Case 1, and we conclude that the probability that $\mathbf{a}_{\mathcal{J}_2} \in \mathsf{Ker}\{\boldsymbol{\theta}_i\}$ is negligible in λ. We thus have $\mathbf{a} \cdot \boldsymbol{\theta}_i \neq 0 \bmod q$ with overwhelming probability, and we can use the same last step as in Case 1 to find the solution of the given instance of the Discrete Logarithm problem. □

4.6 A Variation with Better Efficiency

In the proposed scheme, we could use Vandermonde vectors $\mathbf{a} = (1, a, a^2, \ldots, a^{M-1})$, for some value $a \xleftarrow{R} \mathbb{Z}_q^*$, for the vectors that are included in the secret keys sk_S. The vectors will be linearly independent as long as all the values $a \xleftarrow{R} \mathbb{Z}_q^*$ are different, which happens with overwhelming probability in the security parameter. With this modification, since now the value a is enough to represent the whole vector $\mathbf{a}$, the size of each secret key becomes $|\mathsf{sk}_S| = 1 + |S|$, which is the usual size in the attribute-based literature.

Furthermore, the efficiency of the signature / verification protocols (and the size of the signatures) can be improved in this case, because now the kind of statements in the zero-knowledge proof of knowledge protocol have the form $g^{s_i} = \prod_{1 \leq j \leq M} Y_{i,j}^{a^{j-1}} = g^{f_i(a)}$, where $f_i(Z) = x_{i,1} + x_{i,2}Z + \ldots + x_{i,M}Z^{M-1}$ is the degree $M-1$ polynomial defined by the master secret values associated to attribute at_i. The linear dependence on M in the efficiency of our zero-knowledge proof of knowledge protocol can be reduced to $\sqrt{M}$ or even $\log(M)$, by using the techniques in [21,3] for the polynomial evaluation part of the resulting zero-knowledge proof of knowledge.

The unforgeability proof has to be slightly modified, and in particular a loss factor of $\frac{1}{NL}$ appears in the reduction, since we have to guess, before preparing the public parameters, the indices ℓ^* and i that appear in Case 2 of the proof of Theorem 2.

5 (Bounded) Attribute-Based Encryption: Protocols and Security

Let us move to attribute-based encryption, in both its ciphertext-policy and key-policy flavours. Through the rest of the paper, we will use CP-ABE and KP-ABE as abbreviations of ciphertext-policy attribute-based encryption and key-policy attribute-based encryption.

5.1 CP-ABE: Syntactic Definition

A bounded ciphertext-policy attribute-based encryption (CP-ABE) scheme consists of four probabilistic polynomial-time algorithms:

- $\mathsf{Setup}(1^\lambda)$. The setup algorithm takes as input a security parameter λ and outputs some public parameters pms and a master secret key msk. The public parameters contain the possible universe of attributes $\tilde{\mathcal{P}} = \{\mathsf{at}_1, \ldots, \mathsf{at}_n\}$ and a bound L for the maximum number of users.
- $\mathsf{KeyGen}(S, \mathsf{msk}, \mathsf{pms})$. The key generation algorithm takes as input the master secret key msk, the public parameters pms and then a set of attributes $S \subset \tilde{\mathcal{P}}$ satisfied by the user. The output is a private key sk_S.
- $\mathsf{Encrypt}(m, \mathcal{P}, \Gamma, \mathsf{pms})$. The encryption algorithm takes as input a message m, a decryption policy $(\mathcal{P}, \Gamma)$ where $\mathcal{P} \subset \tilde{\mathcal{P}}$ and $\Gamma \subset 2^{\mathcal{P}}$, and the public parameters pms, and outputs a ciphertext C.
- $\mathsf{Decryption}(C, \mathcal{P}, \Gamma, \mathsf{sk}_S, \mathsf{pms})$. The decryption algorithm takes as input a ciphertext C, a decryption policy $(\mathcal{P}, \Gamma)$, a secret key sk_S and the public parameters pms, and outputs a message $\tilde{m}$.

The usual property of correctness requires that the decryption algorithm, when run on a ciphertext C honestly computed for plaintext m and policy $(\mathcal{P}, \Gamma)$, using secret key sk_S, must output m if and only if $S \cap \mathcal{P} \in \Gamma$.

5.2 CP-ABE: Security Definition

Intuitively, any polinomial-time adversary must have negligible success probability in distinguishing an encryption of $m^{(0)}$ from an encryption of $m^{(1)}$, for the same decryption policy $(\mathcal{P}, \Gamma)$, where the two different plaintexts and the policy are chosen by the adversary. This must hold even if the adversary has adaptive access to an oracle that answers valid secret keys for sets of attributes of his choice, provided none of these subsets is authorized for $(\mathcal{P}, \Gamma)$. This property, usually denoted as IND-CPA, is formally defined via the following experiment $\mathbf{Exp}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{CP}}(\lambda)$,involving an adversary $\mathcal{A}_{CP}$.

$\underline{\mathbf{Exp}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{CP}}(\lambda)}$
Choose $b \xleftarrow{R} \{0,1\}$ at random
$(\mathsf{pms}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$
$(m^{(0)}, m^{(1)}, \mathcal{P}, \Gamma, st_1) \leftarrow \mathcal{A}_{CP}^{\mathsf{KeyGen}(\cdot,\mathsf{msk},\mathsf{pms})}(\mathsf{pms})$
$C^* \leftarrow \mathsf{Encrypt}(m^{(b)}, \mathcal{P}, \Gamma, \mathsf{pms})$
$b' \leftarrow \mathcal{A}_{CP}^{\mathsf{KeyGen}(\cdot,\mathsf{msk},\mathsf{pms})}(C^*, \mathsf{pms}, st_1)$
If some of the following statements is not true, output $\perp$:
(i) $\mathcal{A}_{CP}$ has not made any secret key query S such that $S \cap \mathcal{P} \in \Gamma$;
(ii) the number of secret key queries is at most L (the bound given in pms);
(iii) $m^{(0)} \neq m^{(1)}$
Otherwise, output 1 if $b' = b$, and 0 if $b' \neq b$.

The advantage of $\mathcal{A}_{CP}$ in breaking the bounded IND-CPA property of the CP-ABE scheme is defined as

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{CP}}(\lambda) = \left| \Pr[\mathbf{Exp}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{CP}}(\lambda) = 1] - \frac{1}{2} \right|.$$

Definition 8 A ciphertext-policy attribute-based encryption scheme is indistinguishable under bounded adaptive chosen-plaintext attacks (IND-CPA secure) if, for any adversary $\mathcal{A}_{CP}$ that runs in polynomial time, the advantage $\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{CP}}(\lambda)$ is negligible in the security parameter λ.

5.3 KP-ABE: Syntactic Definition

A bounded key-policy attribute-based encryption (KP-ABE) scheme, for monotone policies, consists of four probabilistic polynomial-time algorithms:

- $\mathsf{Setup}(1^\lambda)$. The setup algorithm takes as input a security parameter λ and outputs some public parameters pms and a master secret key msk. The public parameters contain the possible universe of attributes $\tilde{\mathcal{P}} = \{\mathsf{at}_1, \ldots, \mathsf{at}_n\}$ and a bound L for the maximum number of users.
- $\mathsf{KeyGen}(\mathcal{P}, \Gamma, \mathsf{msk}, \mathsf{pms})$. The key generation algorithm takes as input the master secret key msk, the public parameters pms, a set of attributes $\mathcal{P} \subset \tilde{\mathcal{P}}$ and a monotone policy $\Gamma \subset 2^{\mathcal{P}}$. The output is a private key $\mathsf{sk}_{\mathcal{P},\Gamma}$.

- $\mathsf{Encrypt}(m, S, \mathsf{pms})$. The encryption algorithm takes as input a message m, a set of attributes $S \subset \tilde{\mathcal{P}}$ and the public parameters pms, and outputs a ciphertext C.
- $\mathsf{Decryption}(C, S, \mathsf{sk}_{\mathcal{P},\Gamma}, \mathsf{pms})$. The decryption algorithm takes as input a ciphertext C, the associated set of attributes S, a secret key $\mathsf{sk}_{\mathcal{P},\Gamma}$ and the public parameters pms, and outputs a message $\tilde{m}$.

The correctness property requires that the decryption algorithm, when run on a ciphertext C honestly computed for plaintext m and subset of attributes $S \subset \tilde{\mathcal{P}}$, using secret key $\mathsf{sk}_{\mathcal{P},\Gamma}$, must output m if and only if $S \cap \mathcal{P} \in \Gamma$.

5.4 KP-ABE: Security Definition

Any polinomial-time adversary must have negligible success probability in distinguishing an encryption of $m^{(0)}$ from an encryption of $m^{(1)}$, for the same set of attributes $S \subset \tilde{\mathcal{P}}$, where $m^{(0)} \neq m^{(1)}$ and S are chosen by the adversary. This must hold even if the adversary has adaptive access to an oracle that answers valid secret keys for pairs $(\mathcal{P}, \Gamma)$ of his choice, provided $S \cap \mathcal{P} \notin \Gamma$ holds, for all such queries. This IND-CPA property for KP-ABE schemes is formally defined via the following experiment $\mathbf{Exp}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{KP}}(\lambda)$,involving an adversary $\mathcal{A}_{KP}$.

$\underline{\mathbf{Exp}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{KP}}(\lambda)}$
Choose $b \xleftarrow{R} \{0,1\}$ at random
$(\mathsf{pms}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$
$(m^{(0)}, m^{(1)}, S, st_1) \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\cdot,\mathsf{msk},\mathsf{pms})}_{KP}(\mathsf{pms})$
$C^* \leftarrow \mathsf{Encrypt}(m^{(b)}, S, \mathsf{pms})$
$b' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\cdot,\mathsf{msk},\mathsf{pms})}_{KP}(C^*, \mathsf{pms}, st_1)$
If some of the following statements is not true, output $\perp$:
(i) $\mathcal{A}_{KP}$ has not made any secret key query $(\mathcal{P}, \Gamma)$ such that $S \cap \mathcal{P} \in \Gamma$;
(ii) the number of secret key queries is at most L (the bound given in pms);
(iii) $m^{(0)} \neq m^{(1)}$
Otherwise, output 1 if $b' = b$, and 0 if $b' \neq b$.

The advantage of $\mathcal{A}_{KP}$ in breaking the bounded IND-CPA property of the KP-ABE scheme is defined as

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{KP}}(\lambda) = \left| \Pr[\mathbf{Exp}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{KP}}(\lambda) = 1] - \frac{1}{2} \right|.$$

Definition 9 A key-policy attribute-based encryption scheme is indistinguishable under bounded adaptive chosen-plaintext attacks (IND-CPA secure) if, for any adversary $\mathcal{A}_{KP}$ that runs in polynomial time, the advantage $\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{KP}}(\lambda)$ is negligible in the security parameter λ.

6 The New CP-ABE Scheme (for Threshold Policies)

For simplicity, we consider the case of threshold decryption policies. Therefore, a pair $(\mathcal{P}, \Gamma)$ will be represented as $(\mathcal{P}, t)$, where $1 \leq t \leq |\mathcal{P}|$. Later, we will explain how the scheme can be modified to support other (more general) policies.

$\mathsf{Setup}(1^\lambda)$. The setup algorithm starts by running $(q, \mathbb{G}, g) \leftarrow \mathsf{DLog.Inst}(1^\lambda)$, which outputs a cyclic group $\mathbb{G} = \langle g \rangle$ of prime order q, such that q is λ bits long. The global set of attributes $\tilde{\mathcal{P}} = \{\mathsf{at}_1, \ldots, \mathsf{at}_N\}$ has to be chosen. A bound L for the maximum number of users in the system has to be given. Finally, the value $M = L + 2N$ is defined.

For all $i \in \{1, \ldots, N\}$ and all $j \in \{1, \ldots, M\}$, choose $x_{i,j} \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ independently and at random. Compute $Y_{i,j} = g^{x_{i,j}}$.

The public parameters of the system are $\mathsf{pms} = (q, \mathbb{G}, g, \tilde{\mathcal{P}}, L, N, M, \{Y_{i,j}\}_{1 \leq i \leq N, 1 \leq j \leq M})$, whereas the master secret key is $\mathsf{msk} = \{x_{i,j}\}_{1 \leq i \leq N, 1 \leq j \leq M}$.

$\mathsf{KeyGen}(S, \mathsf{msk}, \mathsf{pms})$. The key generation algorithm takes as input a subset of attributes $S \subset \tilde{\mathcal{P}}$, the master secret key msk and the public parameters pms.

The master entity chooses a vector $\mathbf{a} = (a_1, \ldots, a_M) \stackrel{R}{\leftarrow} (\mathbb{Z}_q)^M$, randomly, from the set of vectors satisfying $\mathbf{a} \cdot (x_{i,1}, \ldots, x_{i,M}) = 1$, for all $\mathsf{at}_i \in S$.

The secret key, $\mathsf{sk}_S = \mathbf{a}$, contains M elements from $\mathbb{Z}_q$.

The correctness of the secret key can be verified by checking that $\prod_{j=1}^{M} Y_{i,j}^{a_j} = g$ holds, for all $\mathsf{at}_i \in S$.

$\mathsf{Encrypt}(m, \mathcal{P}, t, \mathsf{pms})$. The encryption algorithm takes as input a message $m \in \mathbb{G}$, a set of attributes $\mathcal{P} \subset \tilde{\mathcal{P}}$, a threshold t and the public parameters pms. Without loss of generality and to simplify notation, let us assume $\mathcal{P} = \{\mathsf{at}_1, \ldots, \mathsf{at}_n\}$. The algorithm proceeds as follows.

1. Choose at random $r \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$, and compute the value $C_0 = m \cdot g^r$.
2. For $j = 1, \ldots, M$, use Shamir's (t, n)-threshold secret sharing scheme to share 0 among the attributes in $\mathcal{P}$. That is, choose a random polynomial $f_j(x) \in \mathbb{Z}_q[X]$ with degree $t - 1$ such that $f_j(0) = 0$.
3. For each $i = 1, \ldots, n$ and each $j = 1, \ldots, M$, compute the value $C_{i,j} = Y_{i,j}^r \cdot g^{f_j(i)}$.

The final ciphertext, $C = (\, C_0 \,,\, \{C_{i,j}\}_{1 \leq i \leq n, 1 \leq j \leq M} \,)$, contains $nM + 1$ elements from $\mathbb{G}$.

$\mathsf{Decrypt}(C, \mathcal{P}, t, \mathsf{sk}_S, \mathsf{pms})$. The decryption algorithm takes as input a ciphertext C, a set of attributes $\mathcal{P} \subset \tilde{\mathcal{P}}$, a threshold t, a secret key $sk_S = \mathbf{a}$ and the public parameters pms. The algorithm selects a minimally authorized set S', this is, a subset of $S \cap \mathcal{P}$ of cardinality exactly t. Let $\{\lambda_i^{S'}\}_{\mathsf{at}_i \in S'}$ denote the corresponding Lagrange interpolation coefficients, such that $f(0) = \sum_{\mathsf{at}_i \in S'} \lambda_i^{S'} f(i)$, for all polynomial $f(x)$ with degree at most $t - 1$. In particular,

note that for the constant polynomial $f(x) = 1$, we get $1 = \sum_{\mathsf{at}_i \in S'} \lambda_i^{S'}$. The decryption algorithm simply computes and outputs the value

$$C_0 \cdot \prod_{\mathsf{at}_i \in S'} \left(\prod_{j=1}^{M} C_{i,j}^{a_j} \right)^{-\lambda_i^{S'}}$$

6.1 Remarks and Correctness

If some vector $\mathbf{a}$ given to some user as part of his secret key was a linear combination of other vectors $\mathbf{a}^{(\ell)}$ given to other users, then the security of the scheme could be compromised, since the system could not resist a coalition attack. The probability of such a linear dependence is negligible in the security parameter λ. However, in a real implementation of this scheme, it could be desirable to explicitly check that such dependence relations do not occur.

In order to (slightly) improve efficiency, we can partially apply the idea in Section 4.6: vectors $\mathbf{a}$ can be defined as $\mathbf{a} = (\mathbf{a}^{(1)}, \mathbf{a}^{(2)})$, where $\mathbf{a}^{(1)} = (1, a, a^2, \ldots, a^{L-1}) \in (\mathbb{Z}_q)^L$ for some value $a \in \mathbb{Z}_q^*$, and $\mathbf{a}^{(2)} \in (\mathbb{Z}_q)^{2N}$. The first part ensures linear independence, and the second part contains enough degrees of freedom so that $\mathbf{a}$ satisfies the desired properties. In this way, since $\mathbf{a}^{(1)}$ can be represented by element a, the length of $\mathsf{sk}_{\mathcal{P},t}$ becomes $1 + 2N$, independent of the bound L on the number of users.

The correctness of the scheme can be easily verified:

$$C_0 \cdot \prod_{\mathsf{at}_i \in S'} \left(\prod_{j=1}^{M} C_{i,j}^{a_j} \right)^{-\lambda_i^{S'}} = m \cdot g^r \cdot \prod_{\mathsf{at}_i \in S'} \left(\left(\prod_{j=1}^{M} \left(Y_{i,j}^{a_j} \right)^r \right) \cdot \left(\prod_{j=1}^{M} g^{a_j \cdot f_j(i)} \right) \right)^{-\lambda_i^{S'}} =$$

$$m \cdot g^r \cdot \left(\prod_{\mathsf{at}_i \in S'} (g^r)^{-\lambda_i^{S'}} \right) \cdot \prod_{j=1}^{M} \left(\prod_{\mathsf{at}_i \in S'} g^{\lambda_i^{S'} f_j(i)} \right)^{-a_j} = m \cdot g^r \cdot g^{-r} \cdot \prod_{j=1}^{M} \left(g^{f_j(0)} \right)^{-a_j} = m \cdot \prod_{j=1}^{M} 1^{-a_j} = m.$$

6.2 Security of the CP-ABE Scheme

Theorem 3 *Assuming that the Decisional Diffie-Hellman problem is hard in $\mathbb{G}$, then the proposed ciphertext-policy attribute-based encryption scheme is IND-CPA secure.*

Proof Let $\mathcal{A}_{CP}$ be an adversary against the IND-CPA security of the ABE scheme. Without loss of generality, let us assume that:

1. $\mathcal{P} = \{\mathsf{at}_1, \ldots, \mathsf{at}_n\}$ and $t \in \{1, \ldots, n\}$ in the challenge threshold policy $(\mathcal{P}, t)$.

2. $\mathcal{A}_{CP}$ makes L secret key queries, for subsets of attributes $S_1, \ldots, S_L$ such that $|S_\ell \cap \mathcal{P}| = t-1$, for all queries, $\ell = 1, \ldots, L$. Let $\mathsf{sk}_{S_\ell} = \mathbf{a}^{(\ell)}$ denote the obtained secret key, for $\ell = 1, \ldots, L$.

For each attribute $\mathsf{at}_i \in \mathcal{P}$, let us define $\mathcal{Z}_i = \{\mathbf{a}^{(\ell)} \text{ s.t. } \mathsf{at}_i \in S_\ell\}$ and $z_i = |\mathcal{Z}_i|$. We have $0 \leq z_i \leq L$ and $\sum_{1 \leq i \leq n} z_i = (t-1)L$.

Assuming the existence of a successful adversary $\mathcal{A}_{CP}$ with non-negligible advantage $\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{CP}}(\lambda)$, let us construct an adversary $\mathcal{A}_{DDH}$ against the DDH problem. Let (g, g^x, g^y, T) be the given instance of the DDH problem. $\mathcal{A}_{DDH}$ generates public parameters pms for the CP-ABE scheme as follows. $\mathcal{A}_{DDH}$ chooses the global set of attributes $\tilde{\mathcal{P}} = \{\mathsf{at}_1, \ldots, \mathsf{at}_N\}$ and the bound L for the maximum number of users in the system (and so, for the number of extraction queries that $\mathcal{A}$ can make). The value $M = L + 2N$ is defined. To define the values $\{Y_{i,j}\}_{1 \leq i \leq N, 1 \leq j \leq M}$, $\mathcal{A}_{DDH}$ chooses $2N$ vectors $\boldsymbol{\mu}_i = (\mu_{i,1}, \ldots, \mu_{i,M}) \stackrel{R}{\leftarrow} (\mathbb{Z}_q)^M$, $\boldsymbol{\theta}_i = (\theta_{i,1}, \ldots, \theta_{i,M}) \stackrel{R}{\leftarrow} (\mathbb{Z}_q)^M$, for $i = 1, \ldots, N$, all of them independent and random. $\mathcal{A}_{DDH}$ defines $Y_{i,j} = g^{\mu_{i,j}} \cdot (\mathbf{g}^{\mathbf{y}})^{\theta_{i,j}}$, for each $i = 1, \ldots, N$ and each $j = 1, \ldots, M$.

The public parameters of the system that $\mathcal{A}_{DDH}$ gives to $\mathcal{A}_{CP}$ are $\mathsf{pms} = (q, \mathbb{G}, g, \tilde{\mathcal{P}}, L, N, M, \{Y_{i,j}\}_{1 \leq i \leq N, 1 \leq j \leq M})$. During the experiment, $\mathcal{A}_{CP}$ will make up to L extraction queries, to obtain secret keys for subsets S of his choice, where $S \subset \tilde{\mathcal{P}}$. We have to give correct answers to all such queries. To answer the ℓ-th extraction query, where $\ell \in \{1, \ldots, L\}$, corresponding to some subset of attributes S_ℓ, $\mathcal{A}_{DDH}$ chooses $\mathbf{a}^{(\ell)} = (a_1^{(\ell)}, \ldots, a_M^{(\ell)}) \stackrel{R}{\leftarrow} (\mathbb{Z}_q)^M$, randomly from the set of vectors that satisfy $\mathbf{a}^{(\ell)} \cdot \boldsymbol{\mu}_i = 1 \bmod q$ and $\mathbf{a}^{(\ell)} \cdot \boldsymbol{\theta}_i = 0 \bmod q$, for all i such that $\mathsf{at}_i \in S_\ell$, and such that $\mathbf{a}^{(\ell)}$ is linearly independent with $\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(\ell-1)}$.

Since N is an upper bound for the size of S, and the vector space $(\mathbb{Z}_q)^M$ has dimension $M = L + 2N$, it is always possible to find such L linearly independent vectors, and the distribution of the secret keys obtained by $\mathcal{A}_{CP}$ is the same (up to a negligible factor) as in a real execution of the ABE scheme.

At some point $\mathcal{A}_{CP}$ outputs a challenge query for two messages $m^{(0)}, m^{(1)} \in \mathbb{G}$ with $m^{(0)} \neq m^{(1)}$ and a threshold decryption policy $(\mathcal{P}, t)$, where $\mathcal{P} = \{\mathsf{at}_1, \ldots, \mathsf{at}_n\}$ for simplicity. $\mathcal{A}_{DDH}$ chooses at random a bit $b \in \{0, 1\}$ and computes $C_0 = m^{(b)} \cdot (g^x)$. Implicitly, this defines $r = x$ in the challenge ciphertext.

For each $j = 1, \ldots, M$, $\mathcal{A}_{DDH}$ chooses independently and at random a polynomial $f_j(x) = f_j^{(1)} x + \ldots + f_j^{(t-1)} x^{t-1}$ with degree $t-1$ such that $f_j(0) = 0$.

For each $i = 1, \ldots, n$ and each $j = 1, \ldots, M$, $\mathcal{A}_{DDH}$ computes the value

$$C_{i,j} = (g^x)^{\mu_{i,j}} \cdot T^{\theta_{i,j}} \cdot g^{f_j(i)}$$

The challenge ciphertext that $\mathcal{A}_{DDH}$ gives to $\mathcal{A}_{CP}$ is $C^* = (\ C_0\ ,\ \{C_{i,j}\}_{1 \leq i \leq n, 1 \leq j \leq M}\ )$.

When $\mathcal{A}_{CP}$ outputs a bit b', the adversary $\mathcal{A}_{DDH}$ acts as follows: if $b' = b$, then $\mathcal{A}_{DDH}$ concludes that $T = g^{xy}$ and outputs 0; otherwise, if $b' \neq b$, then $\mathcal{A}_{DDH}$ concludes that T is random, and outputs 1.

On the one hand, if the given instance of the DDH problem satisfies $T = g^{xy}$, then it is easy to check that C^* is a valid ciphertext for message $m^{(b)}$. In this case, the output bit b' of $\mathcal{A}_{CP}$ will satisfy $b' = b$ with probability $\frac{1}{2} + \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{CP}}(\lambda)$.

On the other hand, let us discuss what happens when, in the instance of the DDH problem, T is random and independent of x, y. The goal is to show that, in this case, the distribution of the the values that $\mathcal{A}_{CP}$ sees during the attack is independent of the bit b, even if $\mathcal{A}_{CP}$ has unlimited computational power (in particular, we assume that $\mathcal{A}_{CP}$ can compute discrete logarithms, now). If we succeed in proving this, then the output bit b' of $\mathcal{A}_{CP}$ will satisfy $b' = b$ with probability $\frac{1}{2}$ in this case.

Let us write $T = g^{xy+e}$, where $e \neq 0 \bmod q$ with overwhelming probability, in the considered case. The information available to $\mathcal{A}_{CP}$ during the attack includes:

- the discrete logarithms of values $Y_{i,j}$ in pms, for $i = 1, \ldots, n$, $j = 1, \ldots, M$. This means values $\delta_{i,j} := \mu_{i,j} + y\theta_{i,j}$, for $i = 1, \ldots, n$, $j = 1, \ldots, M$;
- the secret keys $\mathsf{sk}_{S_\ell} = \mathbf{a}^{(\ell)}$, satisfying $\mathbf{a}^{(\ell)} \cdot \boldsymbol{\mu}_i = 1$ and $\mathbf{a}^{(\ell)} \cdot \boldsymbol{\theta}_i = 0$, for all $\mathsf{at}_i \in S^{(\ell)}$;
- the discrete logarithms of the elements in the challenge ciphertext C^*. This means $\alpha^{(b)} + \mathbf{x}$, being $m^{(b)} = g^{\alpha^{(b)}}$ for C_0, and finally the discrete logarithm $c_{i,j}$ of each $C_{i,j}$, which is

$$c_{i,j} \;=\; \mathbf{x}(\mu_{i,j} + y\theta_{i,j}) \;+\; e\theta_{i,j} \;+\; \sum_{k=1}^{t-1} f_j^{(k)} i^k$$

  Note that $c_{i,j}$ can be rewriten as $x\delta_{i,j} + e\theta_{i,j} + \sum_{k=1}^{t-1} f_j^{(k)} i^k$.

Let us consider the following vector $\mathbf{X}$ of independent and uniform random variables in $\mathbb{Z}_q$,

$$\mathbf{X} \;=\; \Big(\; x \;,\; \{f_j^{(k)}\}_{1 \le j \le M, 1 \le k \le t-1} \;,\; \{\theta_{i,j}\}_{1 \le i \le n, 1 \le j \le M} \;,\; \{\mu_{i,j}\}_{1 \le i \le n, 1 \le j \le M} \; \Big)$$

and a vector $\mathbf{Y}$ of variables reflecting $\mathcal{A}$'s view (values $c_{i,j}$, $\delta_{i,j}$ and the relations between vectors $\mathbf{a}^{(\ell)}$ and vectors $\boldsymbol{\mu}_i$ and $\boldsymbol{\theta}_i$, for all attribute $\mathsf{at}_i \in S^{(\ell)}$ and $\ell = 1, \ldots, L$):

$$\mathbf{Y} \;=\; \Big(\; x + \alpha^{(b)}, \{c_{i,j}\}_{1 \le i \le n, 1 \le j \le M}, \{\delta_{i,j}\}_{1 \le i \le n, 1 \le j \le M}, 0, \ldots, 0, 1, \ldots, 1 \; \Big)$$

The two vectors of variables are related by an affine transformation $\mathbf{Y} = A \cdot \mathbf{X} + \mathbf{b}$, where $\mathbf{b} = \big(\alpha^{(b)}, 0, \ldots, 0\big)$ is the constant vector, and matrix A has $1 + 2nM + 2L(t-1)$ rows and $1 + (t-1)(M+1) + 2nM$ columns. Since the variables in $\mathbf{X}$ are uniformly random and independent, if we are able to see that the first row of the matrix A is linearly independent from the rest of rows, then we will conclude that the first random variable in $\mathbf{Y}$, which corresponds to element C_0 in the challenge ciphertext, is independent from the other values in $\mathcal{A}$'s view. In such a case, we will conclude that the distribution of the ciphertext is independent of the bit b, as desired.

Lemma 1 *The first row of matrix A is not contained in the vector space spanned by the rest of rows of A.*

Proof To ease the global reading of the paper, in particular of this proof of Theorem 3, the proof of this lemma is moved to Appendix A. □

Final analysis. The probability that $\mathcal{A}_{DDH}$ wins the DDH game is

$$\Pr[\mathbf{Exp}^{\mathsf{ddh}}_{\mathcal{A}_{DDH}}(\lambda)=1] = \Pr[\mathcal{A}_{DDH} \text{ wins}] =$$

$$= \Pr[T=g^{xy}]\cdot\Pr[\mathcal{A}_{DDH} \text{ wins}|T=g^{xy}]+\Pr[T \text{ random}]\cdot\Pr[\mathcal{A}_{DDH} \text{ wins}|T \text{ random}] =$$

$$= \frac{1}{2}\cdot\left(\frac{1}{2}+\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{CP}}(\lambda)\right) \;+\; \frac{1}{2}\cdot\frac{1}{2} \;=\; \frac{1}{2} \;+\; \frac{1}{2}\cdot\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{CP}}(\lambda)$$

Therefore, we have constructed an algorithm $\mathcal{A}_{DDH}$ that solves the DDH problem with non-negligible advantage

$$\mathsf{Adv}^{\mathsf{ddh}}_{\mathcal{A}_{DDH}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{ddh}}_{\mathcal{A}_{DDH}}(\lambda)=1]-\frac{1}{2}\right| = \frac{1}{2}\cdot\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{CP}}(\lambda).$$

□

7 The New KP-ABE Scheme (for Threshold Policies)

Again, we consider for the moment the case of threshold policies. Therefore, a pair $(\mathcal{P},\Gamma)$ will be represented as $(\mathcal{P},t)$, where $1\leq t\leq|\mathcal{P}|$. The new KP-ABE scheme is very similar to the CP-ABE scheme in the previous section.

$\mathsf{Setup}(1^\lambda)$. The setup algorithm starts by running $(q,\mathbb{G},g)\leftarrow\mathsf{DLog.Inst}(1^\lambda)$, which outputs a cyclic group $\mathbb{G}=\langle g\rangle$ of prime order q, such that q is λ bits long. The global set of attributes $\tilde{\mathcal{P}}=\{\mathsf{at}_1,\ldots,\mathsf{at}_N\}$ has to be chosen. A bound L for the maximum number of users in the system has to be given. Finally, the value $M=L+2N$ is defined.

For all $i\in\{1,\ldots,N\}$ and all $j\in\{1,\ldots,M\}$, choose $x_{i,j}\stackrel{R}{\leftarrow}\mathbb{Z}_q^*$ independently and at random. Compute $Y_{i,j}=g^{x_{i,j}}$.

The public parameters of the system are $\mathsf{pms}=(q,\mathbb{G},g,\tilde{\mathcal{P}},L,N,M,\{Y_{i,j}\}_{1\leq i\leq N,1\leq j\leq M})$, whereas the master secret key is $\mathsf{msk}=\{x_{i,j}\}_{1\leq i\leq N,1\leq j\leq M}$.

$\mathsf{KeyGen}(\mathcal{P},t,\mathsf{msk},\mathsf{pms})$. The key generation algorithm takes as input a subset of n attributes $\mathcal{P}\subset\tilde{\mathcal{P}}$ and a threshold t such that $1\leq t\leq n\leq N$, the master secret key msk and the public parameters pms.

The master entity uses Shamir's (t,n)-threshold secret sharing scheme to share 1 among the attributes in $\mathcal{P}$. That is, he chooses a random polynomial $f(x)\in\mathbb{Z}_q[X]$ with degree $t-1$ such that $f(0)=1$, and he defines $s_i=f(i)$, for all $\mathsf{at}_i\in\mathcal{P}$.

After that, the master entity chooses a vector $\mathbf{a}=(a_1,\ldots,a_M)\stackrel{R}{\leftarrow}(\mathbb{Z}_q)^M$, randomly, from the set of vectors satisfying $\mathbf{a}\cdot(x_{i,1},\ldots,x_{i,M})=s_i$, for all $\mathsf{at}_i\in\mathcal{P}$.

The secret key, $\mathsf{sk}_{\mathcal{P},t} = \mathbf{a}$, contains M elements from $\mathbb{Z}_q$.

The correctness of the secret key can be verified by checking that, for any subset $S \subset \mathcal{P}$ with $|S| = t$, it holds $\prod_{\mathsf{at}_i \in S} \left(\prod_{j=1}^{M} Y_{i,j}^{a_j} \right)^{\lambda_i^S} = g$, where $\{\lambda_i^S\}_{\mathsf{at}_i \in S}$ denote the corresponding Lagrange interpolation coefficients.

$\mathsf{Encrypt}(m, S, \mathsf{pms})$. The encryption algorithm takes as input a message $m \in \mathbb{G}$, a set of attributes $S \subset \tilde{\mathcal{P}}$ and the public parameters pms. Let us assume, for simplicity, that $S = \{\mathsf{at}_1, \ldots, \mathsf{at}_n\}$, with $|S| = n \leq N$. The algorithm proceeds as follows.

1. Choose at random $r \xleftarrow{R} \mathbb{Z}_q^*$, and compute the value $C_0 = m \cdot g^r$.
2. For each $i = 1, \ldots, n$ and each $j = 1, \ldots, M$, compute the value $C_{i,j} = Y_{i,j}^r$.

The final ciphertext, $C = (C_0 , \{C_{i,j}\}_{1 \leq i \leq n, 1 \leq j \leq M})$, contains $nM + 1$ elements from $\mathbb{G}$.

$\mathsf{Decrypt}(C, S, \mathsf{sk}_S, \mathsf{pms})$. The decryption algorithm takes as input a ciphertext C, the associated set of attributes $S \subset \tilde{\mathcal{P}}$, a secret key $sk_{\mathcal{P},t} = \mathbf{a}$ and the public parameters pms. The algorithm selects a minimally authorized set $S' \subset S \cap \mathcal{P}$ of cardinality exactly t. Let $\{\lambda_i^{S'}\}_{\mathsf{at}_i \in S'}$ denote the corresponding Lagrange interpolation coefficients, such that $f(0) = \sum_{\mathsf{at}_i \in S'} \lambda_i^{S'} f(i)$, for all polynomial $f(x)$ with degree at most $t-1$. In particular, note that for the constant polynomial $f(x) = 1$, we get $1 = \sum_{\mathsf{at}_i \in S'} \lambda_i^{S'}$. The decryption algorithm simply computes and outputs the value

$$C_0 \cdot \prod_{\mathsf{at}_i \in S'} \left(\prod_{j=1}^{M} C_{i,j}^{a_j} \right)^{-\lambda_i^{S'}}$$

7.1 Remarks and Correctness

The same remarks as in the case of the CP-ABE scheme can be made now, regarding the linear independence of the vectors $\mathbf{a}$ in different secret keys, and the efficiency improvement (regarding the size of the secret keys) by considering $\mathbf{a} = (1, a, a^2, \ldots, a^{L-1}, \mathbf{a}^{(2)})$ for some value $a \in \mathbb{Z}_q^*$ and some vector $\mathbf{a}^{(2)} \in (\mathbb{Z}_q)^{2N}$.

The correctness of the KP-ABE scheme can be easily verified:

$$C_0 \cdot \prod_{\mathsf{at}_i \in S'} \left(\prod_{j=1}^{M} C_{i,j}^{a_j} \right)^{-\lambda_i^{S'}} = m \cdot g^r \cdot \prod_{\mathsf{at}_i \in S'} \left(\prod_{j=1}^{M} \left(Y_{i,j}^{a_j} \right)^r \right)^{-\lambda_i^{S'}} =$$

$$m \cdot g^r \cdot \prod_{\mathsf{at}_i \in S'} \left(g^{s_i}\right)^{-r \cdot \lambda_i^{S'}} = m \cdot g^r \cdot \left(g^{\sum_{\mathsf{at}_i \in S'} \lambda_i^{S'} f(i)} \right)^{-r} = m \cdot g^r \cdot \left(g^1\right)^{-r} = m.$$

7.2 Security of the KP-ABE Scheme

Theorem 4 *Assuming that the Decisional Diffie-Hellman problem is hard in $\mathbb{G}$, then the proposed key-policy attribute-based encryption scheme is IND-CPA secure.*

Proof (Sketch.) The proof follows the same ideas as the proof of Theorem 3 for the security of the CP-ABE scheme. We transform a hypothetical successful adversary $\mathcal{A}_{KP}$ against the KP-ABE scheme into an adversary $\mathcal{A}_{DDH}$ against the DDH problem. If (g, g^x, g^y, T) is the given instance of the DDH problem, $\mathcal{A}_{DDH}$ generates the public parameters pms for the KP-ABE scheme exactly as in the proof of Theorem 3. In particular, $\mathcal{A}_{DDH}$ defines $Y_{i,j} = g^{\mu_{i,j}} \cdot (\mathbf{g^y})^{\theta_{i,j}}$, for each $i = 1, \ldots, N$ and each $j = 1, \ldots, M$.

When $\mathcal{A}_{KP}$ makes its ℓ-th extraction query, to obtain a secret key for a pair $(\mathcal{P}_\ell, t_\ell)$ of its choice, $\mathcal{A}_{DDH}$ chooses at random a degree $t_\ell - 1$ polynomial $f^{(\ell)}(x)$ such that $f^{(\ell)}(0) = 1$, and then chooses a vector $\mathbf{a}^{(\ell)} = (a_1^{(\ell)}, \ldots, a_M^{(\ell)}) \stackrel{R}{\leftarrow} (\mathbb{Z}_q)^M$, randomly from the set of vectors that satisfy $\mathbf{a}^{(\ell)} \cdot \boldsymbol{\mu}_i = f^{(\ell)}(i) \bmod q$ and $\mathbf{a}^{(\ell)} \cdot \boldsymbol{\theta}_i = 0 \bmod q$, for all i such that $\mathsf{at}_i \in \mathcal{P}_\ell$, and such that $\mathbf{a}^{(\ell)}$ is linearly independent with $\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(\ell-1)}$.

Since N is an upper bound for the size of S, and the vector space $(\mathbb{Z}_q)^M$ has dimension $M = L + 2N$, it is always possible to find such L linearly independent vectors, whose distribution is essentially the same as in a real execution of the KP-ABE scheme.

When $\mathcal{A}_{KP}$ outputs a challenge query for two messages $m^{(0)}, m^{(1)} \in \mathbb{G}$ with $m^{(0)} \neq m^{(1)}$ and a subset of attributes $S \subset \tilde{\mathcal{P}}$, $\mathcal{A}_{DDH}$ chooses at random a bit $b \in \{0, 1\}$ and computes $C_0 = m^{(b)} \cdot (g^x)$. For each $i = 1, \ldots, n$ and each $j = 1, \ldots, M$, $\mathcal{A}_{DDH}$ computes the value $C_{i,j} = (g^x)^{\mu_{i,j}} \cdot T^{\theta_{i,j}}$.

When $\mathcal{A}_{KP}$ outputs a bit b', the adversary $\mathcal{A}_{DDH}$ acts as follows: if $b' = b$, then $\mathcal{A}_{DDH}$ concludes that $T = g^{xy}$ and outputs 0; otherwise, if $b' \neq b$, then $\mathcal{A}_{DDH}$ concludes that T is random, and outputs 1.

If the given instance of the DDH problem satisfies $T = g^{xy}$, then C^* is a valid ciphertext for message $m^{(b)}$. In this case, the output bit b' of $\mathcal{A}_{KP}$ will satisfy $b' = b$ with probability $\frac{1}{2} + \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{KP}}(\lambda)$.

On the other hand, when T is random and independent of x, y in the given instance of the DDH problem, then we can again write $T = g^{xy+e}$ for a random value e, different from zero with overwhelming probability. In this case, we can show (by using similar arguments as in the proof of Theorem 3) that the distribution of the the values that $\mathcal{A}_{KP}$ sees during the attack is independent of the bit b. Note that $C_0 = m^{(b)} \cdot (g^x)$, and the dependence on x of all the other elements $C_{i,j}$ in the challenge ciphertext is cancelled by the randomness $\theta_{i,j}$ in values $Y_{i,j}$ of the public parameters. Some degrees of freedom/independence on these values $\theta_{i,j}$ are "lost" in front of $\mathcal{A}_{KP}$ when

he makes the L extraction queries for pairs $(\mathcal{P}_\ell, t_\ell)$; note however that these queries must satisfy $|\mathcal{P}_\ell \cap S| \leq t_\ell - 1$, by definition, and that there are precisely $t_\ell - 1$ perfectly hidden degrees of freedom in the choice of the polynomial $f^{(\ell)}(x)$. Therefore, and informally speaking (this can be formally proved, as we have done in the proof of Theorem 3) the $t_\ell - 1$ degrees of freedom on values $\{\theta_{i,j}\}_{\mathsf{at}_i \in S, 1 \leq j \leq M}$ that are "lost" by the conditions $\boldsymbol{\theta}_i \cdot \mathbf{a}^{(\ell)} = 0$ for all $\mathsf{at}_i \in \mathcal{P}_\ell \cap S$ are compensated by the $t_\ell - 1$ perfectly hidden values $\{f^{(\ell)}(i)\}_{\mathsf{at}_i \in \mathcal{P}_\ell \cap S}$ that univocally determine, along with $f^{(\ell)}(0) = 1$, the polynomial $f^{(\ell)}(x)$.

Therefore, the conclusion is the same as in the proof of Theorem 3: the existence of an algorithm $\mathcal{A}_{DDH}$ that solves the DDH problem with non-negligible advantage

$$\mathsf{Adv}^{\mathsf{ddh}}_{\mathcal{A}_{DDH}}(\lambda) = \frac{1}{2} \cdot \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{A}_{KP}}(\lambda).$$

□

8 Extensions, (In)Efficiency Considerations and Applications

8.1 More General Policies

In the description of our schemes, we have considered for simplicity the particular case where policies are of the threshold family. However, the schemes can be extended to admit more general (monotone) policies $\Gamma \subset \mathcal{P}$. In the attribute-based signature scheme, in particular in the zero-knowledge proof of knowledge protocol, the idea is to consider a secret sharing scheme (for instance, a monotone span program [23]) that realizes the dual access structure $\Gamma^* = \{S \subset \mathcal{P} \mid \mathcal{P} - S \notin \Gamma\}$; the value c will be the secret, and the values $\{c_j\}_{\mathsf{at}_j \in \mathcal{P}}$ will be the shares, that will be computed with the secret sharing scheme for the access structure Γ^*. In the threshold case, the dual access structure is $\Gamma^* = \{S \subset \mathcal{P} \mid |S| \geq n - t + 1\}$, and the secret is shared with Shamir's method, with a polynomial of degree $n - t$. This kind of zero-knowledge proofs were introduced for the first time in [13]. Apart from replacing polynomials with monotone span programs, the rest of the protocol and the security analysis work exactly in the same way as we detail here for the threshold case.

In the attribute-based encryption schemes, the idea is simply to replace polynomials with monotone span programs that realize the corresponding policy Γ. Another difference is that some (algebraic) parts of the security proofs become (even) more tedious.

8.2 (In)Efficiency

The fact that a bound L on the total number of users must be fixed from the beginning is a drawback, but in some real-life applications this may not be a serious limitation. However, the fact that the efficiency of the new schemes (size of the public parameters, secret keys, signatures, ciphertexts, running times...)

depend on L is a more serious drawback, which may limit the applicability of our schemes to very restricted scenarios, like small companies or institutions.

Elements in our signatures or ciphertexts may belong to a standard group $\mathbb{G}$ where the discrete logarithm problem is hard (points of an elliptic curve) and can thus be securely represented by $\lambda = 160$ bits, in contrast to the the 1024 or 2048 bits needed to represent elements in RSA-based [22] or pairing-based [25] solutions. But this "advantage" quickly disappears since the number of elements in our (non-optimized) signatures or ciphertexts is $n(6+L+N)$ or $n(L+2N)$, where N is the total number of attributes in the system. In the other schemes in the literature, for general monotone policies and with adaptive security, this number is typically $6n$ or $9n$. Focusing on the encryption case, this means that our schemes can be competitive only for values $L + 2N \leq 70$, for instance in applications with $N = 20$ attributes and $L = 30$ users.

Since we do not want to finish the work with a negative opinion of the new schemes, we show in the next section that the bounded-security property of our schemes (in particular, the KP-ABE scheme) is enough to find a positive application.

8.3 Application to Verifiable Computation

Parno, Raykova and Vaikuntanathan give in [30] a general construction from a KP-ABE scheme to a protocol for the verifiable delegation of computation of boolean functions $f : \{0,1\}^N \rightarrow \{0,1\}$. Note that a boolean function is equivalent to an access structure on a set of N elements (or attributes, in the context of this paper).

The basic idea is that the client runs the setup phase of the KP-ABE scheme twice, $(\mathsf{pms}_0, \mathsf{msk}_0) \leftarrow \mathsf{KP\text{-}ABE.Setup}(1^\lambda)$ and $(\mathsf{pms}_1, \mathsf{msk}_1) \leftarrow \mathsf{KP\text{-}ABE.Setup}(1^\lambda)$. Later, to delegate the computation of some boolean function f admitted by the KP-ABE scheme, the client runs $\mathsf{sk}_f \leftarrow \mathsf{KP\text{-}ABE.KeyGen}(f, \mathsf{msk}_0, \mathsf{pms}_0)$ and $\mathsf{sk}_{\bar{f}} \leftarrow \mathsf{KP\text{-}ABE.KeyGen}(\bar{f}, \mathsf{msk}_1, \mathsf{pms}_1)$, where $\bar{f}$ is the negation of f: $\bar{f}(\mathbf{x}) = 1 - f(\mathbf{x})$, for all $\mathbf{x} \in \{0,1\}^N$. The values $(\mathsf{pms}_0, \mathsf{pms}_1, \mathsf{sk}_f, \mathsf{sk}_{\bar{f}})$ are given to the server. The client chooses a secure one-way function H.

Each time the client wants the server to compute $f(\mathbf{x})$ on some boolean vector $\mathbf{x} \in \{0,1\}^N$, he chooses two random plaintexts $m^{(0)}, m^{(1)}$ and computes ciphertexts $C^{(0)} \leftarrow \mathsf{KP\text{-}ABE.Encrypt}(m^{(0)}, \mathbf{x}, \mathsf{pms}_0)$ and $C^{(1)} \leftarrow \mathsf{KP\text{-}ABE.Encrypt}(m^{(1)}, \mathbf{x}, \mathsf{pms}_1)$, along with the hash values $s_0 = H(m^{(0)})$ and $s_1 = H(m^{(1)})$. The clieint sends $\mathbf{x}$ and the ciphertexts $C^{(0)}, C^{(1)}$ to the server, and stores or publishes the pair (s_0, s_1).

If $f(\mathbf{x}) = 1$, the server computes $m^{(0)} \leftarrow \mathsf{KP\text{-}ABE.Decrypt}(C^{(0)}, \mathbf{x}, \mathsf{sk}_f, \mathsf{pms}_0)$ and sends $(m^{(0)}, \perp)$ to the client. If $f(\mathbf{x}) = 0$ (and so $\bar{f}(\mathbf{x}) = 1$), the server computes $m^{(1)} \leftarrow \mathsf{KP\text{-}ABE.Decrypt}(C^{(1)}, \mathbf{x}, \mathsf{sk}_{\bar{f}}, \mathsf{pms}_1)$ and sends $(\perp, m^{(1)})$ to the client.

Depending on the obtained answer and its relation to the stored hashed values (s_0, s_1), the client (or anybody) concludes what is the correct value of $f(\mathbf{x})$. Furthermore, thanks to the two parallel executions of the KP-ABE

process, the server cannot cheat the client without being detected; the worse he can do is to abort the process. That is, the computation performed by the server is (publicly) verifiable.

Parno et al. prove in [30] that the resulting protocol for (publicly) verifiable computation of boolean functions is secure, provided the underlying KP-ABE scheme is *one-key secure*, which means that it must be secure in front of adversaries that can make at most one secret key extraction query. Translated to the framework of this paper, we can use our bounded KP-ABE scheme with $L = 1$ (and thus, $M = 1+2N$) to instantiate the Parno et al. construction. As a result, we obtain a protocol for the verifiable computation (VC) of monotone boolean functions, in the standard Discrete Logarithm scenario, with security based on the DDH Assumption. A (small) drawback of the resulting VC protocol with respect to other instantiations of the construction, for instance using pairing-based KP-ABE schemes, is that the cost for the client is quadratic on N, rather than linear, because the encryption running time in our KP-ABE scheme depends on $NM = 2N^2 + N$.

8.3.1 A Comment on the Monotonicity of f (and $\bar{f}$).

As we have noted in the previous paragraph, the resulting VC protocol would be valid for monotone boolean functions, because the underlying KP-ABE scheme only admits monotone functions (or access policies). This is not a serious problem in practice; as noted in [30], a non-monotone boolean function f_1 can always be transformed into an equivalent monotone one, by doubling the number N of variables (including the negation of each variable). Note that this observation would be important even in the case where the VC protocol was run to evaluate a monotone (increasing) function f, because the underlying KP-ABE scheme must admit the function $\bar{f}$, which is monotone decreasing. If the KP-ABE scheme admits only monotone increasing functions, then f must not be modified, but we should find a monotone increasing function equivalent to $\bar{f}$.

This means that, with the transformation proposed in [30] applied to a KP-ABE scheme which admits only monotone increasing functions, one always needs to double the number of variables, from N to $2N$, even if the client only wants to evaluate monotone increasing functions. This observation is important since most of the KP-ABE schemes proposed in the literature (including the new one, in this work) only admit monotone increasing functions; a notable exception can be found in [29].

However, it turns out that the transformation in [30] can be modified in the following way. First of all, in the delegation phase, replace $\mathsf{sk}_{\bar{f}} \leftarrow \mathsf{KP\text{-}ABE.KeyGen}(\bar{f}, \mathsf{msk}_1, \mathsf{pms}_1)$ with $\mathsf{sk}_{f^*} \leftarrow \mathsf{KP\text{-}ABE.KeyGen}(f^*, \mathsf{msk}_1, \mathsf{pms}_1)$, where f^* is the dual function of f: $f^*(\mathbf{x}) = 1 - f(\bar{\mathbf{x}})$, for all $\mathbf{x} \in \{0,1\}^N$. Later, in the request phase, replace $C^{(1)} \leftarrow \mathsf{Encrypt}(m^{(1)}, \mathbf{x}, \mathsf{pms}_1)$ with $C^{(1)} \leftarrow \mathsf{Encrypt}(m^{(1)}, \bar{\mathbf{x}}, \mathsf{pms}_1)$. With these modifications, the server can either obtain $m^{(0)}$, if $f(\mathbf{x}) = 1$, or obtain $m^{(1)}$, if $f^*(\bar{\mathbf{x}}) = 1$, which is equivalent to $f(\mathbf{x}) = 0$. So the functionality of the construction is preserved, but now we have that

the monotonicity of f^* is the same as that of f (if any). Therefore, if a client is interested in the verifiable computation of a monotone increasing function f, and the underlying KP-ABE scheme admits only this kind of functions, we can run this modified version of the Parno et al. construction without doubling the number of variables.

References

1. M. Abdalla, F. Bourse, A. De Caro and D. Pointcheval. Simple functional encryption schemes for inner products. *Proc. of PKC'15*, LNCS **9020**, Springer-Verlag, pp. 733–751 (2015).
2. F. Baldimtsi and A. Lysyanskaya. Anonymous credentials light. *Proc. of CCS'13*, ACM Press, pp. 1087–1098 (2013)
3. S. Bayer and J. Groth. Zero-knowledge argument for polynomial evaluation with application to blacklists. *Proc. of Eurocrypt'13*, LNCS **7881**, Springer-Verlag, pp. 646–663 (2013)
4. M. Bellare and G. Fuchsbauer. Policy-based signatures. *Proc. of PKC'14*, LNCS **8383**, Springer-Verlag, pp. 520–537 (2014)
5. M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. *Proc. of CCS'93*, ACM Press, pp. 62–73 (1993)
6. M. Bellare and S. Shoup. Two-tier signatures, strongly unforgeable signatures, and Fiat-Shamir without random oracles. *Proc. of PKC'07*, LNCS **4450**, Springer-Verlag, pp. 201–216 (2007)
7. J. Bethencourt, A. Sahai and B. Waters. Ciphertext-policy attribute-based encryption. *Proc. of IEEE Symposium on Security and Privacy*, IEEE Society Press, pp. 321–334 (2007)
8. D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan and D. Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. *Proc. of Eurocrypt'14*, LNCS **8441**, Springer-Verlag, pp. 533–556 (2014)
9. X. Boyen. Attribute-based functional encryption on lattices. *Proc. of TCC'13*, LNCS **7785**, Springer-Verlag, pp. 122–142 (2013)
10. S. Brands. Rapid demonstration of linear relations connected by boolean operators. *Proc. of Eurocrypt'97*, LNCS **1233**, Springer-Verlag, pp. 318–333 (1997)
11. D. Chaum, E. van Heijst and B. Pfitzmann. Cryptographically strong undeniable signatures, unconditionally secure for the signer. *Proc. of Crypto'91*, LNCS **576**, Springer-Verlag, pp. 470–484 (1992)
12. D. Chaum and T.P. Pedersen. Wallet databases with observers. *Proc. of Crypto'92*, LNCS **740**, Springer-Verlag, pp. 89–105 (1993)
13. R. Cramer, I. Damgård, B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. *Proc. of Crypto'94*, LNCS **839**, Springer-Verlag, pp. 174–187 (1994)
14. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, Vol. **31**, pp. 469–472 (1985)
15. A. Escala, J. Herranz and P. Morillo. Revocable attribute-based signatures with adaptive security in the standard model. *Proc. of Africacrypt'11*, LNCS **6737**, Springer-Verlag, pp. 224–241 (2011)
16. U. Feige and A. Shamir. Witness indistinguishable and witness hiding protocols. *Proc. of STOC'90*, ACM Press, pp. 416–426 (1990)
17. A. Fiat and A. Shamir. How to prove yourself: practical solutions to identification and signature problems. *Proc. of Crypto'86*, LNCS **263**, Springer-Verlag, pp. 186–194 (1986)
18. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *Proc. of FOCS'13*, IEEE Society Press, pp. 40–49 (2013)

19. S. Garg, C. Gentry, S. Halevi, A. Sahai and B. Waters. Attribute-based encryption for circuits from multilinear maps. *Proc. of Crypto'13*, LNCS **8043**, Springer-Verlag, pp. 479–499 (2013)
20. V. Goyal, O. Pandey, A. Sahai and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. *Proc. of Computer and Communications Security, CCS'06*, ACM Press, pp. 89–98 (2006)
21. J. Groth. Linear algebra with sub-linear zero-knowledge arguments. *Proc. of Crypto'09*, LNCS **5677**, Springer-Verlag, pp. 192–208 (2009)
22. J. Herranz. Attribute-based signatures from RSA. *Theoretical Computer Science*, Vol. **527**, pp. 73–82 (2014)
23. M. Karchmer and A. Wigderson. On span programs. *Proc. of SCTC'93*, IEEE Computer Society Press, pp. 102–111 (1993)
24. A. Lewko, T. Okamoto, A. Sahai, K. Takashima and B. Waters. Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption. *Proc. of Eurocrypt'10*, LNCS **6110**, Springer-Verlag, pp. 62–91 (2010)
25. A. Lewko and B. Waters. New proof methods for attribute-based encryption: achieving full security through selective techniques. *Proc. of Crypto'12*, LNCS **7417**, Springer-Verlag, pp. 180–198 (2012)
26. H.K. Maji, M. Prabhakaran and M. Rosulek. Attribute-based signatures. *Proc. of CT-RSA'11*, LNCS **6558**, Springer-Verlag, pp. 376–392 (2011)
27. P- Mohassel. One-time signatures and chameleon hash functions. *Proc. of SAC'10*, LNCS **6544**, Springer-Verlag, pp. 302–319 (2010)
28. T. Okamoto and K. Takashima. Efficient attribute-based signatures for non-monotone predicates in the standard model. *Proc. of PKC'11*, LNCS **6571**, Springer-Verlag, pp. 35–52 (2011)
29. R. Ostrovsky, A. Sahai and B. Waters. Attribute-based encryption with non-monotonic access structures. *Proc. of CCS'07*, ACM Press, pp. 195–203 (2007)
30. B. Parno, M. Raykova and V. Vaikuntanathan. How to delegate and verify in public: verifiable computation from attribute-based encryption. *Proc. of TCC'12*, LNCS **7194**, Springer-Verlag, pp. 422–439 (2012)
31. D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, Vol. **13** (3), pp. 361–396 (2000)
32. J. Pieprzyk, H. Wang and C. Xing. Multiple-time signature schemes against chosen message attacks. *Proc. of SAC'03*, LNCS **3006**, Springer-Verlag, pp. 88–100 (2003)
33. C.P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, Vol. **4**, Springer-Verlag, pp. 161–174 (1991)

A Proof of Lemma 1

The first row of A is $(1, 0, \ldots, 0)$. The following nM rows, corresponding to values $c_{i,j}$ in vector $\mathbf{Y}$, reflect the relation

$$c_{i,j} = x\delta_{i,j} + e\theta_{i,j} + \sum_{k=1}^{t-1} f_j^{(k)} i^k$$

The following nM rows, corresponding to values $\delta_{i,j}$ in $\mathcal{A}$'s view, reflect the relation

$$\delta_{i,j} = y\theta_{i,j} + \mu_{i,j}.$$

The following $L(t-1)$ rows reflect the relations

$$\mathbf{a}^{(\ell)} \cdot \boldsymbol{\theta}_i = 0,\ \forall \mathsf{at}_i \in S^{(\ell)},\ \forall \ell = 1, \ldots, L.$$

Finally, the last $L(t-1)$ rows of matrix A reflect the relations

$$\mathbf{a}^{(\ell)} \cdot \boldsymbol{\mu}_i = 1,\ \forall \mathsf{at}_i \in S^{(\ell)},\ \forall \ell = 1, \ldots, L.$$

The general form of matrix A is shown in Figure 1.

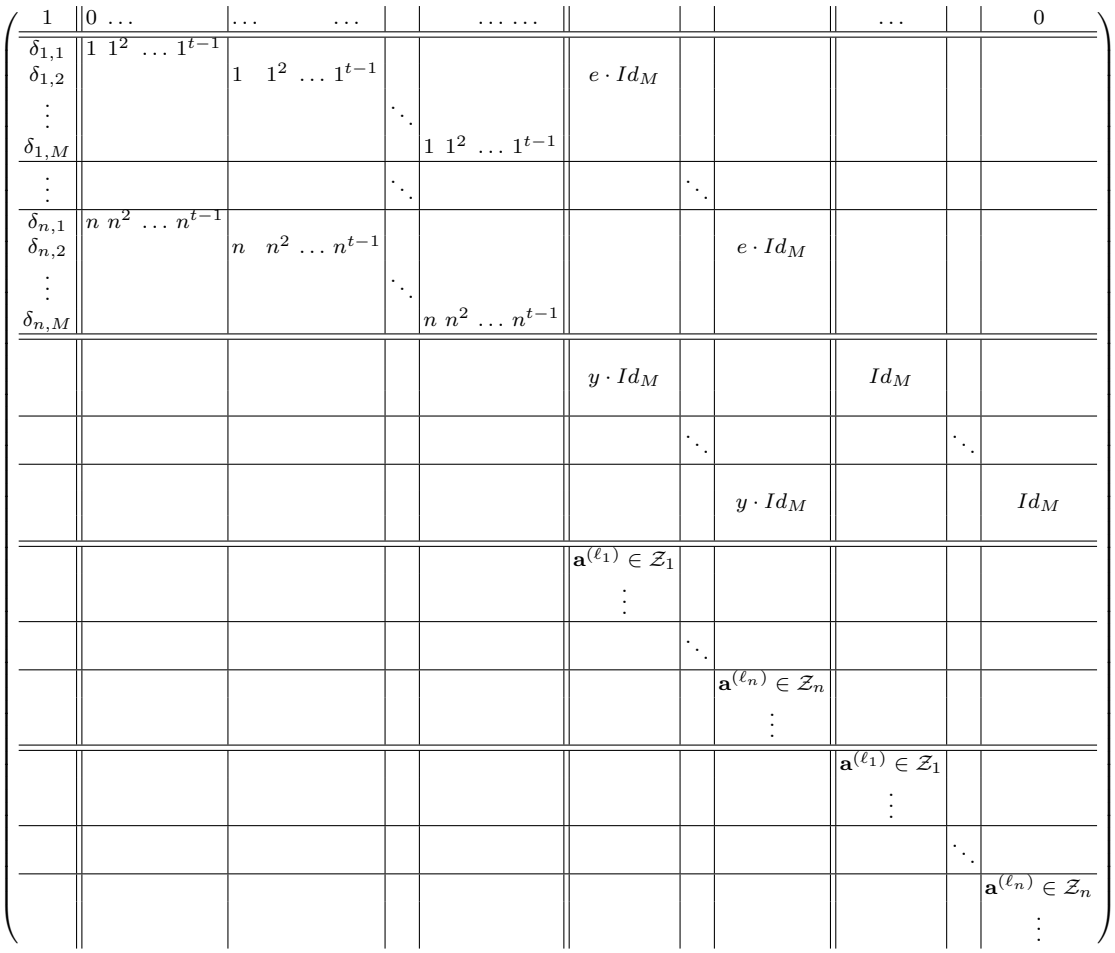

Fig. 1: The initial matrix, A

The last n blocks of rows in A can be substracted with linear combinations of the rows in the blocks with $y \cdot Id_M\ \dots\ Id_M$, and we get the transformed but equivalent matrix $A^{(2)}$, depicted in Figure 2.

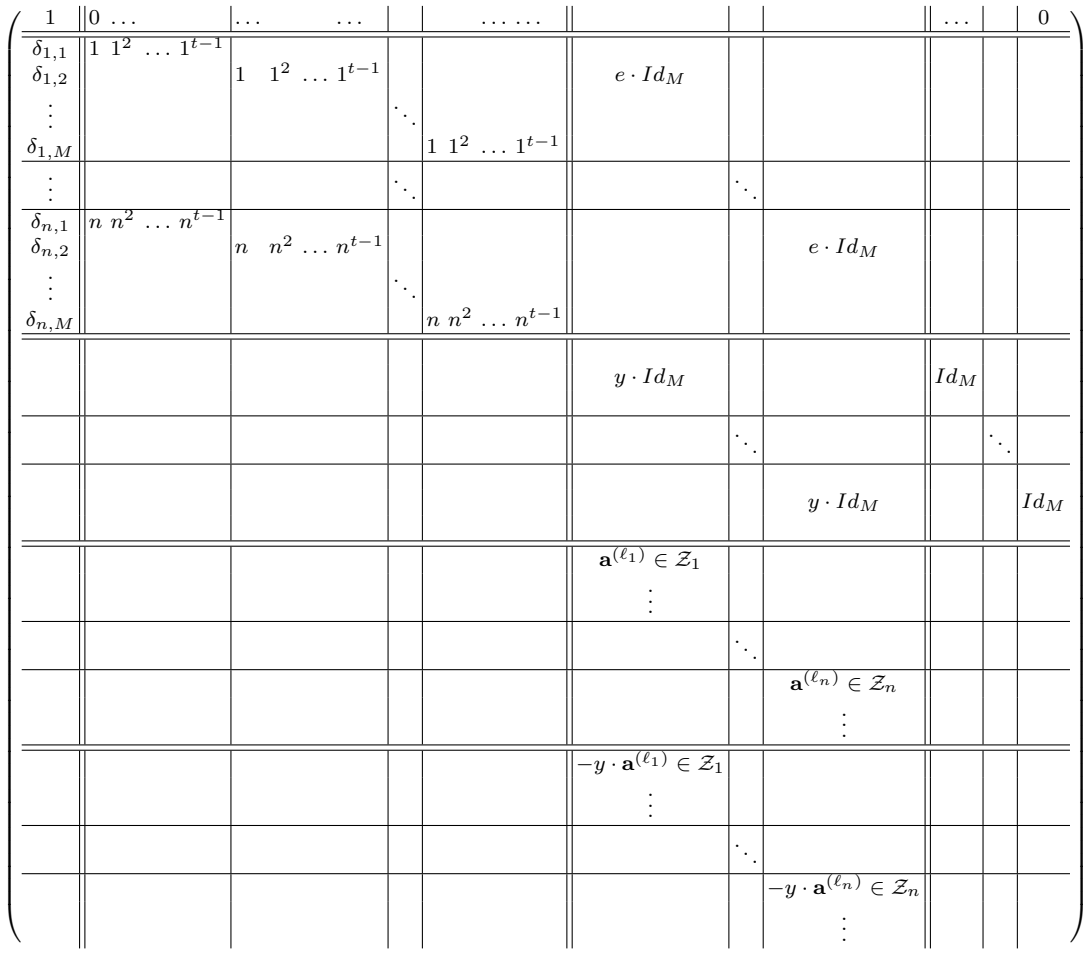

Fig. 2: Matrix $A^{(2)}$

Since our goal is to prove that the first row of the matrix is not spanned by the rest of rows of the matrix, we can now remove the rows and columns "touched" by the blocks Id_M on the right part of the matrix, because the coefficients of these rows in a hypothetical linear combination of all the rows that would equal the vector $(1, 0, \dots, 0)$ would be equal to 0. Similarly, we can remove the last n blocks of rows, because these rows are multiples of the rows in the previous n blocks (by multiplying them with $-y$).

All in all, we get a reduced matrix $A^{(3)}$, in Figure 3, where we still want to show that the first row is not spanned by the rest of rows.

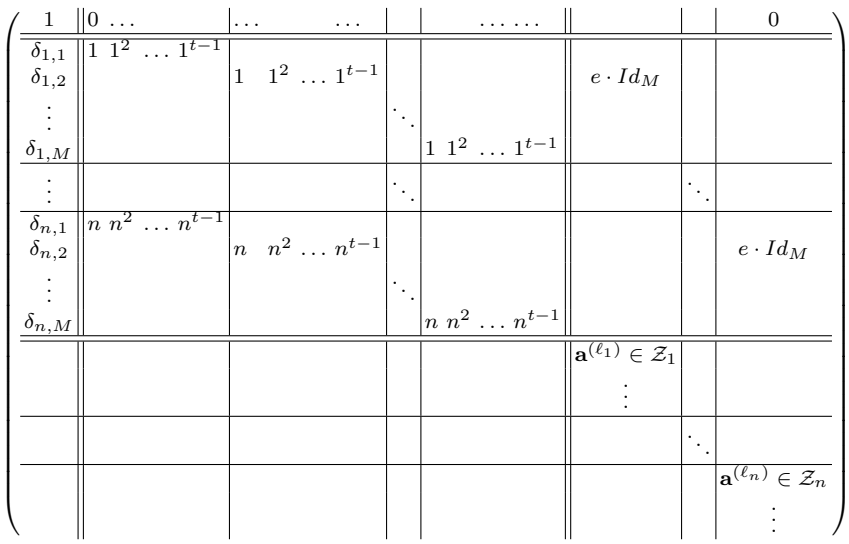

Fig. 3: Matrix $A^{(3)}$

In the M central blocks of columns of $A^{(3)}$, with Vandermonde vectors, we can reorder the rows in order to get $(1, \dots, 1^{t-1})$, and below $(2, \dots, 2^{t-1})$, and so on, until $(t-1, \dots, (t-1)^{t-1})$, and then the rest of vectors until $(n \dots, n^{t-1})$ and the same effect repeated M times, in diagonal descending cascade. We can transform the rows corresponding to each index $s \in \{t, t+1, \dots, n\}$ by substracting to them linear combinations (with the Lagrange interpolation coefficients) of the rows corresponding to indices in $\{1, \dots, t-1\}$, in order to get 0's in all these $n - t + 1$ rows of each of those M blocks. We denote as $\lambda_i^{(s)}$ the corresponding Lagrange coefficients, such that $(s, \dots, s^{t-1}) = \sum_{1 \leq i \leq t-1} \lambda_i^{(s)} (i, \dots, i^{t-1})$.

What we get after applying these transformations is matrix $A^{(4)}$, in Figure 4. Values marked with $*$ in $A^{(4)}$ and the following matrices are not relevant.

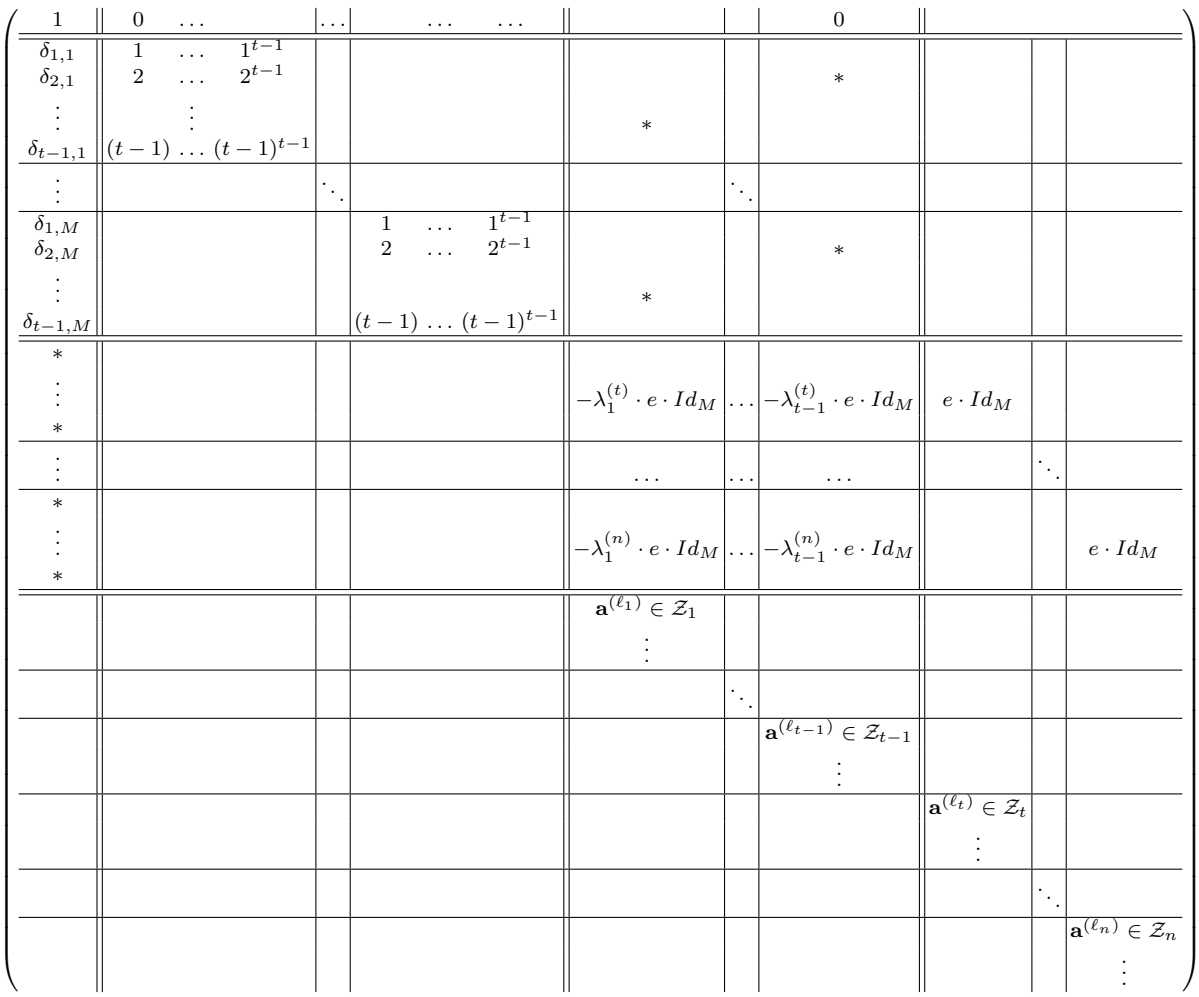

Fig. 4: Matrix $A^{(4)}$

Now the non-zero rows of the central M blocks of columns are clearly linearly independent (seen as vectors in $(\mathbb{Z}_q)^{M(t-1)}$) so, again, a hypothetical linear combination of all the rows in $A^{(4)}$ (excepting the first one) that would equal the vector $(1, 0, \ldots, 0)$ would have 0 as the coefficient of these rows. We can thus remove the corresponding rows and columns, and we get matrix $A^{(5)}$, in Figure 5, where we still want to pove that the first row is not spanned by the rest of rows.

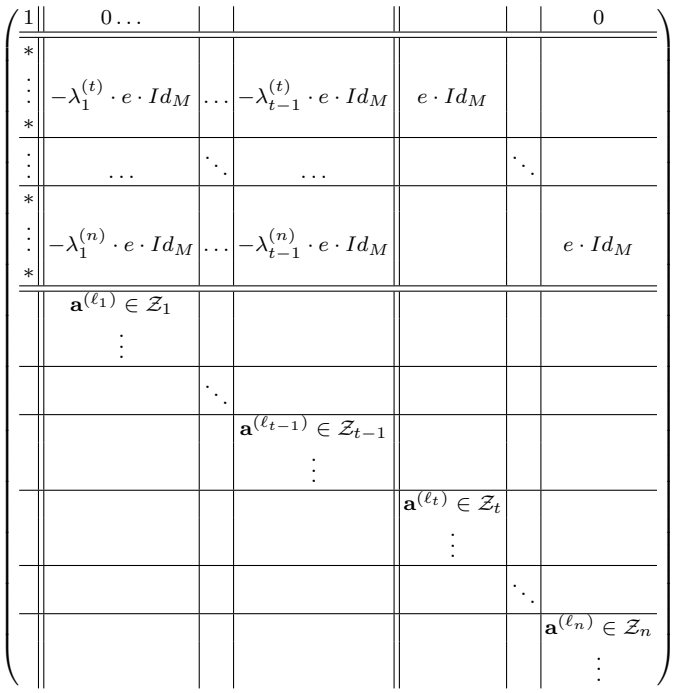

Fig. 5: Matrix $A^{(5)}$

In the next transformation, we are going to transform the last $z_t + \ldots + z_n$ rows of $A^{(5)}$ into 0, by substracting to them some linear combinations of the rows with $e \cdot Id_M$ on the right part. As a result, we will be able to remove all the columns and rows "touched" by those blocks $e \cdot Id_M$, because these rows cannot contribute in a hypothetical linear combination of all the rows in the modified $A^{(5)}$ matrix (excepting the first one) that would equal the vector $(1, 0, \ldots, 0)$. The result of these two steps is matrix $A^{(6)}$, depicted in Figure 6.

$$\left(\begin{array}{c||c|c|c|c}
1 & 0\ldots & 0\ldots & 0\ldots & \ldots 0 \\
\hline\hline
 & \mathbf{a}^{(\ell_1)} \in \mathcal{Z}_1 & & & \\
 & \vdots & & & \\
\hline
 & & \mathbf{a}^{(\ell_2)} \in \mathcal{Z}_2 & & \\
 & & \vdots & & \\
\hline
 & & & \ddots & \\
\hline
 & & & & \mathbf{a}^{(\ell_{t-1})} \in \mathcal{Z}_{t-1} \\
 & & & & \vdots \\
\hline
* & \lambda_1^{(t)} \cdot \mathbf{a}^{(\ell_t)} \in \mathcal{Z}_t & \lambda_2^{(t)} \cdot \mathbf{a}^{(\ell_t)} \in \mathcal{Z}_t & \ldots & \lambda_{t-1}^{(t)} \cdot \mathbf{a}^{(\ell_t)} \in \mathcal{Z}_t \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
\hline
\vdots & \vdots & \vdots & \ddots & \vdots \\
\hline
* & \lambda_1^{(n)} \cdot \mathbf{a}^{(\ell_n)} \in \mathcal{Z}_n & \lambda_2^{(n)} \cdot \mathbf{a}^{(\ell_n)} \in \mathcal{Z}_n & \ldots & \lambda_{t-1}^{(n)} \cdot \mathbf{a}^{(\ell_n)} \in \mathcal{Z}_n \\
\vdots & \vdots & \vdots & \vdots & \vdots
\end{array}\right)$$

Fig. 6: Matrix $A^{(6)}$

Let us now look for possible linear combinations of the rows (all but the first one) of matrix $A^{(6)}$ that could equal the first row, that is, vector $(1, 0, \ldots, 0)$. Let us denote the coefficients of such a hypothetical linear combination as $\{\rho_i^{(\ell)}\}_{1 \leq \ell \leq L, \mathsf{at}_i \in S_\ell \cap \mathcal{P}}$.

Looking at the right part, below the 0's of the first row, and taking into account that these vectors are all linearly independent, we have that the sum of the coefficients of all the rows corresponding to a same vector $\mathbf{a}^{(\ell)}$ must be 0. Let us focus on one such vector $\mathbf{a}^{(\ell)}$, corresponding to a secret key query for a subset of attributes S_ℓ such that $|S_\ell \cap \mathcal{P}| = t - 1$. Let us define $A_\ell = S_\ell \cap \{\mathsf{at}_1, \ldots, \mathsf{at}_{t-1}\}$ and $B_\ell = S_\ell \cap \{\mathsf{at}_t, \ldots, \mathsf{at}_n\}$, and let us denote $r = |A_\ell|$ and $w = |B_\ell|$, such that $t - 1 = r + w$. The involved coefficients $\{\rho_i^{(\ell)}\}_{\mathsf{at}_i \in S_\ell \cap \mathcal{P}}$ must satisfy the following conditions, for all $i = 1, \ldots, t-1$ (that is, for each of the $t-1$ blocks of M columns below the 0's of the first row):

(i) if $\mathsf{at}_i \in A_\ell$, then $\rho_i^{(\ell)} + \sum_{\mathsf{at}_s \in B_\ell} \rho_s^{(\ell)} \cdot \lambda_i^{(s)} = 0$,

(ii) if $\mathsf{at}_i \notin A_\ell$, then $\sum_{\mathsf{at}_s \in B_\ell} \rho_s^{(\ell)} \cdot \lambda_i^{(s)} = 0$.

In other words, the vector of coefficients $(\{\rho_i^{(\ell)}\}_{\mathsf{at}_i \in A_\ell}, \{\rho_s^{(\ell)}\}_{\mathsf{at}_s \in B_\ell})$ must be in the kernel of the $(t-1) \times (t-1)$ matrix

$$D = \left(\begin{array}{c|c}
 & \lambda_{i_1}^{(s_1)} \ \ldots \ \lambda_{i_1}^{(s_w)} \\
 & \vdots \quad\quad \vdots \\
Id_r & \lambda_{i_r}^{(s_1)} \ \ldots \ \lambda_{i_r}^{(s_w)} \\
\hline
 & \lambda_{i_{r+1}}^{(s_1)} \ \ldots \ \lambda_{i_{r+1}}^{(s_w)} \\
 & \vdots \quad\quad \vdots \\
 & \lambda_{i_{t-1}}^{(s_1)} \ \ldots \ \lambda_{i_{t-1}}^{(s_w)}
\end{array}\right)$$

where $A_\ell = \{\mathsf{at}_{i_1}, \ldots, \mathsf{at}_{i_r}\}$, $B_\ell = \{\mathsf{at}_{s_1}, \ldots, \mathsf{at}_{s_w}\}$ and we use $i_{r+1}, \ldots, i_{t-1}$ to refer to the indices in $\{1, \ldots, t-1\} - A_\ell$. The bottom-right part of D is a non-singular $w \times w$ matrix, because it is a square submatrix of the $(t-1) \times (t-1)$ matrix Λ of Lagrange interpolation

coefficients that one would obtain by considering $t-1$ interpolation points $\{1,\ldots,t-1\}$ and $t-1$ external evaluation points $\{s_1,\ldots,s_w,\tilde{s}_{w+1},\ldots,\tilde{s}_{t-1}\}$. Matrix Λ is non-singular because it is the product of two non-singular matrices: the matrix which has the coefficients of the $t-1$ interpolation polynomials (a basis) as rows, and the matrix which has the Vandermonde vectors of the evaluation points as columns.

Since we have the identity matrix Id_r on the top-left part of D, we conclude that D is non-singular, and therefore there is no vector in the kernel of D, other than the zero vector $\mathbf{0}$.

Summing up, for all the vectors $\mathbf{a}^{(\ell)}$, $\ell = 1,\ldots,L$, we have that all the coefficients $\{\rho_i^{(\ell)}\}_{\mathsf{at}_i \in S_\ell \cap \mathcal{P}}$ must be equal to 0. This concludes the proof of the fact that the first row of matrix A is not spanned by the rest of the rows of A.