Multi-Authority ABE for Non-Monotonic Access Structures

Miguel Ambrona[1] and Romain Gay[2]

[1] Nomadic Labs
[2] IBM Research Zurich

Abstract. Attribute-Based Encryption (ABE) is a cryptographic primitive which supports fine-grained access control on encrypted data, making it an appealing building block for many applications. Multi-Authority Attribute-Based Encryption (MA-ABE) is a generalization of ABE where the central authority is distributed across several independent parties. We provide the first MA-ABE scheme from asymmetric prime-order pairings where no trusted setup is needed and where the attribute universe of each authority is unbounded. Moreover, it is the first to handle non-monotonic access structures. These features broaden the applicability and improve the efficiency of our scheme. Our construction makes a modular use of Functional Encryption schemes with fine-grained access control.

1 Introduction

Attribute-Based Encryption (ABE) [SW05, GPSW06] subsumes traditional public-key encryption by providing fine-grained access to the encrypted data. Namely, each ciphertext is associated with an access policy, and each user receives a so-called user secret key according to their credentials. If these credentials fulfill the policy, the user secret key can be used to successfully decrypt the ciphertext. Otherwise, the plaintext remains hidden, even if several non-authorized users collude.

Despite being a prominent topic in the research community, the notion of ABE suffers from several drawbacks. User secret keys are generated from a so-called master secret key, which can decrypt any ciphertext. Consequently, the generation of these keys must be performed by a trusted third party, who controls the master secret key and who must be online every time a key is requested (not only during the setup phase of the scheme). Such a third party is a single point of failure in the system and is likely to be a target for attacks. Copying the master secret and using redundant servers to alleviate this bottleneck only increases the chances of key exposure. Besides, the master secret key owner can impersonate any user of its choice, acting as an escrow (see [Rog15] for further details on this issue). To mitigate these shortcomings, a solution is to decentralize the key-generation so that no single party holds the master secret key in full. Furthermore, decentralization is encouraged given that in many scenarios the access policy used to generate a ciphertext includes attributes coming from different organizations.

The work of [Cha07] and later [MKE08] considered a variation of ABE where any party can become an authority by publishing some public key; these authorities, created on the fly, handle different attributes, and no coordination is required among them. In these systems, a user equipped with a global identifier can collect different credentials associated with different attributes from each authority. However, the user must then interact with a trusted central authority that will process such credentials and provide the actual ABE user secret keys. The advantage of their approach is that this central authority is agnostic to the meaning of the attributes and credentials of the user, and does not need to communicate with the other authorities. However, most of the aforementioned shortcomings remain. Afterward, [LCLS08] removed the need for a central authority, but the set of authorities in their construction is fixed and they must interact during the setup phase. Another limitation is that the security of their scheme is only proven for an a priori bounded number of collusions. [CC09] also presented a scheme with no central authority relying on distributed PRFs. However, their scheme is still limited in terms of expressiveness (it can only express a strict AND policy) and only handles a pre-determined set of attributes. In [LW11], the authors gave the first construction where there is no central authority, authorities can join the system on the fly without communicating with each other and the ciphertexts can be associated with a rich class of expressive access policies (including Boolean formulas). Despite these impressive features, their construction still suffers from some limitations: it requires a trusted setup; it uses inefficient composite-order pairings; each authority can only handle a small (poly-size) set of attributes as, in fact, the public key of each authority grows with the number of attributes owned by the authority. Later on, in [OT13, RW15], the authors built MA-ABE where there is no trusted setup beyond the mere agreement of which groups and which hash function to use, and where the attribute set of each authority is of exponential size or unbounded. Moreover, these schemes have the advantage of using prime-order pairings, which are more efficient than their composite-order counterparts. However, the scheme from [OT13] is not shown to achieve security in the presence of corrupted authorities, an important requirement in the standard security definition for MA-ABE. The scheme from [RW15] inherits from [LW11] prohibitively large ciphertexts. Indeed, in these schemes, each ciphertext contains a number of *target* group elements that grows with the size of its associated access policy, which are significantly larger than source group elements. Another reason all existing schemes lack practical efficiency is their use of *symmetric* pairings, which are less efficient than their asymmetric counterparts. This is in contrast with state-of-the-art single-authority ABE schemes, defined over *asymmetric* pairings and without target group elements in the ciphertext.

Finally, existing MA-ABE can only handle monotonic access structures. Namely, policies that can be expressed by a Boolean formula with *positive literals* only, e.g. of the form: $\mathsf{Role} = \mathit{Reviewer} \wedge \mathsf{Year} = \mathit{2022}$. Suppose the document to be encrypted is an audit of the security department of some company for the year 2022. In order to avoid conflict of interests, employees from the security department should not be able to access the document. This corresponds to the non-monotonic formula: $(\mathsf{Role} = \textit{Reviewer} \wedge \mathsf{Year} = \textit{2022}) \wedge \neg(\mathsf{Department} = \textit{Security})$. A naive way to implement negative literals would be to give user secret keys associated with both the credential owned by the user and all the negative literals not owned by the user, e.g. $\neg(\mathsf{HumanResources})$, $\neg(\mathsf{IT})$, $\neg(\mathsf{Marketing})$, $\neg(\mathsf{R\&D})$, $\neg(\mathsf{Production})$, and so on, for all existing departments in the company where the user does not belong. This solution yields very large user secret keys, since they grow proportionally with the number of possible attributes. In fact, this becomes unfeasible for large attribute universe (where the number of attribute is super-polynomial), let alone unbounded universe (where there is no restriction on the number of possible attributes, i.e. any bit string can serve as an attribute). [OSW07] gave the first ABE for non-monotonic formulas, but their techniques do not seem to be directly applicable to the multi-authority setting. This prompts the question: *Can we achieve MA-ABE with similar features and efficiency than single-authority ABE?*

Our contribution. We provide the first MA-ABE scheme from asymmetric prime-order pairings, without trusted setup and where the attribute universe of each authority is of unbounded size. Furthermore, our scheme handles non-monotonic access structures. It makes a modular use of practical Functional Encryption (FE) schemes for simple functions, namely, inner-products (we refer to our technical overview for more details about the FE we use). We prove security from standard assumptions using pairings (namely, the SXDH assumption) in the random oracle model. Our construction achieves security against adversaries that can choose the access structure of the challenge ciphertext and the attributes of the user secret keys, but the access structure and the attributes chosen cannot depend on the cryptographic material received. That is, they must not depend on the challenge ciphertext or the user secret keys (although they can depend on the public key). We refer to this security notion as super-selective security — the selective security notion traditionally refers to the setting where the adversary is constrained to choose the access structure used in the challenge ciphertext before receiving any cryptographic material (either the public key or the user secret keys). We leave it as an open problem to obtain adaptive security. Table 1 compares our scheme with the state-of-the-art.

Technical overview. We consider an MA-ABE where access policies are represented by monotone span programs (MSP) (as per Definition 1), which capture monotonic Boolean formulas. We explain how to handle non-monotonic formulas later in this overview. In a nutshell, an MSP allows users to produce shares $s_1, \ldots, s_\ell$ of a secret s, where ℓ is the size of the MSP, and each share s_j is associated with an attribute $\rho(j)$. Akin to standard secret sharing schemes, the secret s can be recovered if and only if sufficiently many shares s_j are given. The ABE uses cyclic groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}$ of prime order p, equipped with a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_\mathsf{t}$. We use additive bracket notation for all three groups,

Reference	[LW11]	[RW15]	[OT13]	This work
pairing type	comp. sym.	prime. sym.	prime sym.	prime asym.
assumption	composite	q-type	DLIN	SXDH
security	adaptive	selective	adaptive	s-selective
attribute universe	small	large	large	large
attributes per authority	bounded	unbounded	bounded	unbounded
non-monotonic access structures	no	no	no	yes
corrupted authorities	yes	yes	no	yes

Table 1. Comparison among MA-ABE schemes. The attribute universe is said to be small when it is a-priori bounded by a polynomial in the security parameter. It is said to be large when it is of a-priori bounded exponential size in the security parameters, or not bounded at all. "Corrupted authorities" refers to whether or not the scheme is secure when the adversary can acquire the secret keys of some authorities of their choice, or even create authorities with a public key of their choice (this is the standard definition for MA-ABE). "q-type" refers to a family of parameterized computational assumptions in pairing groups. "s-selective" refers to the super-selective security notion (defined in Section 2.5).

that is, for $\mathsf{s} \in \{1, 2, \mathsf{t}\}$, and all scalars $x \in \mathbb{Z}_p$, we write $[\![x]\!]_{\mathsf{s}} = xP_{\mathsf{s}}$ where P_{s} is a generator of $\mathbb{G}_{\mathsf{s}}$. Finally, we make use of Functional Encryption (FE) schemes, which are an advanced form of public-key encryption where the secret key can be used to derive functional secret keys sk_f for certain functions f. Decryption can use sk_f to extract from an encryption $\mathsf{Enc}(\mathsf{pk}, m)$ of the message m the value $f(m)$. Nothing else is revealed about the message m apart from the value $f(m)$. Many functional secret keys can be derived for different functions from the secret key (which is referred to as master secret key, just like in the ABE setting). In short, FE enables selective computations on encrypted data. We rely on practical FE schemes that handle a particular class of functions of interest.

For encryption, an exponent s is uniformly sampled from $\mathbb{Z}_p$ and the encapsulation key is defined as $[\![s]\!]_{\mathsf{t}}$ (we consider the KEM variant of ABE). The MSP is used to create shares $\{s_j\}_{j\in[\ell]}$ of s and shares $\{u_j\}_{j\in[\ell]}$ of 0. The MA-ABE ciphertext consists of one FE ciphertext of the vector (s_j, u_j) per $j \in [\ell]$. The public key of the FE used for each $j \in [\ell]$ is published by the authority that owns the attribute $\rho(j)$. Note that in order to register into the system, each authority will run the FE setup algorithm to create its pair of keys $(\mathsf{FE.pk}, \mathsf{FE.msk})$.

The FE we are using is for identity-based inner-products. That is, each ciphertext encrypts a vector $\boldsymbol{x}$ (of some fixed dimension, say d, which is then set to 2 for our modular construction), and an identity id. Each functional secret key is associated with a vector $[\![\boldsymbol{y}]\!]_2 \in \mathbb{G}_2^d$ and an identity id'. The decryption of the ciphertext with the functional secret key succeeds if the identities match, in which case the inner-product $[\![\boldsymbol{x}^\top \boldsymbol{y}]\!]_{\mathsf{t}}$ is recovered. Nothing else is revealed about the encrypted vector $\boldsymbol{x}$. However, we do not require that the identities id and id' or the vector $[\![\boldsymbol{y}]\!]_2$ remain hidden. These functional secret keys can be generated from the master secret key of the FE scheme.

As we explained, the MA-ABE ciphertext will contain the FE encryption of the vector (s_j, u_j) for the identity $\rho(j)$, under the $\mathsf{FE.pk}$ of the authority that owns attribute $\rho(j)$, for all $j \in [\ell]$.

The secret key of a user identified by a global identifier gid, for an attribute att, will contain the FE functional secret key for the vector $[\![(1, z_{\mathsf{gid}})]\!]_2$ and the identity att, where $[\![z_{\mathsf{gid}}]\!]_2$ is the output of the hash value $H(\mathsf{gid})$. This FE functional secret key is computed using the FE master secret key of the authority that owns the attribute att.

The user gid collects all the FE functional secret keys $\mathsf{sk}_{[\![(1,z_{\mathsf{gid}})]\!]_2,\mathsf{att}}$, by making a request $(\mathsf{att}, \mathsf{gid})$ to the relevant authorities. Each FE key $\mathsf{sk}_{[\![(1,z_{\mathsf{gid}})]\!]_2,\mathsf{att}}$ yields the value $[\![s_j + z_{\mathsf{gid}} u_j]\!]_{\mathsf{t}}$ if j is such that $\rho(j) = \mathsf{att}$. If sufficiently many such values are revealed, then they can be combined to obtain $[\![s + z_{\mathsf{gid}} \cdot 0]\!]_{\mathsf{t}} = [\![s]\!]_{\mathsf{t}}$, the encapsulation key. Otherwise said, if the user gid possesses enough attributes to satisfy the MSP in the ciphertext, it recovers the encapsulation key. Here we rely on the fact that the share reconstruction for an MSP is linear.

To argue security, we could simply rely on the simulation security of the underlying FE scheme, which states that only the value $[\![s_j + z_{\mathsf{gid}} u_j]\!]_{\mathsf{t}}$ is revealed by the ciphertext and the FE functional secret key for identity $\rho(j)$ and vector $[\![(1, z_{\mathsf{gid}})]\!]_2$ (together with the value $[\![z_{\mathsf{gid}}]\!]_2$, which is public). Note that the term $[\![z_{\mathsf{gid}} u_j]\!]_{\mathsf{t}}$ prevent collusions across different gid. In fact it hides the share s_j, assuming the values $[\![z_{\mathsf{gid}}]\!]_2$ generated by the hash function are pseudo-random (this holds in the random oracle model). So, if for any given gid there are not enough attributes to satisfy the access structure associated to the ciphertext, then there are not enough values $[\![s_j + z_{\mathsf{gid}} u_j]\!]_{\mathsf{t}}$ to recover $[\![s]\!]_{\mathsf{t}}$, which remains hidden.

This approach works, but it requires an FE scheme that is simulation-secure with many challenge ciphertexts. Unfortunately, such primitive cannot be built from standard assumptions (this can be proved by an incompressibility argument, similar to [BSW11], see Remark 1). We use an FE with indistiguishability-based security instead, which means that our MA-ABE requires a more sophisticated security proof relying on some prime-order variant of the dual vector pairing space methodology [OT09, Lew12]. Our modular construction can be instantiated with any FE with indistinguishability-based security for the appropriate functionality, such as the scheme from [ACGU20].

We now explain how to handle non-monotonic access structures, represented by span programs where each share is associated with either a normal or negated attribute (as per Definition 2). For negated attributes, we simply replace the identity-based FE for inner-products (which we call $\mathcal{FE}_1$ here) in our modular construction with an FE with revocations (called $\mathcal{FE}_2$). That is, the ciphertext of $\mathcal{FE}_2$ encrypts a vector $\boldsymbol{x}$ together with an identity id, as before, but now each functional secret key is associated with a vector $[\![\boldsymbol{y}]\!]_2$ and a set of identities $\mathcal{S}$. If $\mathsf{id} \notin \mathcal{S}$, then the decryption recovers $[\![\boldsymbol{x}^\top \boldsymbol{y}]\!]_\mathsf{t}$. Else, no information is revealed about $\boldsymbol{x}$ (although the identity id, the vector $[\![\boldsymbol{y}]\!]_2$ and the set $\mathcal{S}$ are not hidden). We present a new construction for such an FE scheme whose selective security is proven under standard pairing assumptions (SXDH). Our modular MA-ABE for non-monotonic access structures uses $\mathcal{FE}_1$ and $\mathcal{FE}_2$ as follows. The encryption creates shares $\{s_j\}_{j\in[\ell]}$ of a random value s and shares $\{u_j\}_{j\in[\ell]}$ of 0 according to the span program that represents the access structure, as before. The novelty here is that each share $j \in [\ell]$ is mapped to $\rho(j)$ which is either a normal attribute, in which case the encryption encrypts the vector (s_j, u_j) with the identity $\rho(j)$ using $\mathcal{FE}_1$; or it is mapped to $\rho(j)$ which is a negated attribute, in which case the encryption encrypts (s_j, u_j) with the identity $\rho(j)$ but this time using $\mathcal{FE}_2$. Let gid be the global identifier of a user that possesses different sets of attributes $\mathcal{S}_\mathsf{aut} = \{\mathsf{att}_1^\mathsf{aut}, \ldots, \mathsf{att}_{n_\mathsf{aut}}^\mathsf{aut}\}$ each owned by a different authority aut. For each authority aut, the user collects the $\mathcal{FE}_2$ functional secret key for the vector $[\![(1, z_\mathsf{gid})]\!]_2$ and the set $\mathcal{S}_\mathsf{aut}$, together with a set of n_aut $\mathcal{FE}_1$ functional secret keys for the vector $[\![(1, z_\mathsf{gid})]\!]_2$ and the identity $\mathsf{att}_i^\mathsf{aut}$ for $i = 1, \ldots, n_\mathsf{aut}$. Thanks to these keys, a user can recover the values $[\![s_j + z_\mathsf{gid} u_j]\!]_\mathsf{t}$ for the shares j associated with $\rho(j)$ which is either a normal attribute owned by the user gid, or a negated attributed that is not part of the set of attributes owned by gid. As a result, decryption succeeds if and only if the attributes of the user gid satisfy the non-monotonic access structure. The security of the MA-ABE boils down to the security of the underlying FE schemes.

To build the FE for inner-products with revocations, we start with a one-time statistically secure scheme where the encryption of a vector $\boldsymbol{x} \in \mathbb{Z}_p^n$ for an identity $\mathsf{id}^\star \in \mathbb{Z}_p$ is of the form $\mathsf{ct} = \big(\boldsymbol{x} + \boldsymbol{v}, P(\mathsf{id}^\star)\big)$ where $\boldsymbol{v} \in \mathbb{Z}_p^n$ is a random vector and P is a random polynomial evaluated on $\mathsf{id}^\star \in \mathbb{Z}_p$. The functional secret key for a vector $\boldsymbol{y} \in \mathbb{Z}_p^n$ and a set of identities $\mathcal{S} \subset \mathbb{Z}_p$ is of the form $\mathsf{sk}_{\boldsymbol{y},\mathcal{S}} = \big(\boldsymbol{y}^\top \boldsymbol{v} + P(0), \{P(\mathsf{id})\}_{\mathsf{id}\in\mathcal{S}}\big)$. We assume the identity space is $\mathbb{Z}_p^*$, excluding 0 as a valid identity. Polynomial P is of degree d, and we assume the set $\mathcal{S}$ associated with each functional secret key is of size exactly d. We explain later how to remove this restriction. If $\mathsf{id}^\star \notin \mathcal{S}$, we have the evaluation of the polynomial P on $d+1$ distinct points, so we can recover $P(0)$ using Lagrange interpolation and get $\boldsymbol{y}^\top \boldsymbol{v}$, thanks to which we can obtain $\boldsymbol{x}^\top \boldsymbol{y}$. On the other hand, if $\mathsf{id}^\star \in \mathcal{S}$, we only have the evaluation of P on d distinct points, which reveals no information about $P(0)$, which completely masks $\boldsymbol{v}^\top \boldsymbol{y}$. Therefore, $\boldsymbol{v}$ masks $\boldsymbol{x}$ perfectly. To obtain an FE scheme with public-key encryption and security for many functional secret keys, we use standard techniques from pairing groups:

- instead of using the vector $\boldsymbol{v}$ and the polynomial P, the encryption uses $[\![\boldsymbol{v}]\!]_1$ and the coefficients of P in $\mathbb{G}_1$ that are part of pk to compute:

$$\mathsf{ct} = \left([\![\boldsymbol{x} + \boldsymbol{v}r]\!]_1, [\![rP(\mathsf{id}^\star)]\!]_1\right) \ , \text{ for } r \leftarrow_R \mathbb{Z}_p \ .$$

- to obtain security against collusions, we randomize the functional secret keys:

$$\mathsf{sk}_{\boldsymbol{y},\mathcal{S}} = \left([\![\boldsymbol{y}^\top \boldsymbol{v} + sP(0)]\!]_2, \{[\![sP(\mathsf{id})]\!]_2\}_{\mathsf{id}\in\mathcal{S}}\right) \ , \text{ for } s \leftarrow_R \mathbb{Z}_p \ .$$

The scheme describe here would be secure in the generic group model. To accommodate for a security proof using the SXDH assumption (i.e. the assumption that DDH holds both in $\mathbb{G}_1$ and $\mathbb{G}_2$), we modify slightly the scheme using techniques reminiscent from the hash proof system from [CS02], similarly to [ALS16] in the context of functional encryption for inner-products.

Remark 1 (Impossibility of simulation secure FE). We consider an adversary playing against the many-ciphertexts simulation security of an identity-based FE scheme for inner-products, which makes q_1 functional secret key queries for random vectors $[\![\boldsymbol{y}_1]\!]_2, \ldots, [\![\boldsymbol{y}_{q_1}]\!]_2$ and identities $\mathsf{id}_1, \ldots, \mathsf{id}_{q_1}$. The adversary also chooses random vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{q_2}$ and identities $\mathsf{id}^\star_1, \ldots, \mathsf{id}^\star_{q_2}$ for the challenge ciphertexts. The adversary chooses $\mathsf{id}_1 = \mathsf{id}_2 = \cdots = \mathsf{id}_{q_1} = \mathsf{id}^\star_1 = \cdots = \mathsf{id}^\star_{q_2}$. The simulator must produce the challenge ciphertexts and the functional secret keys using only the values $[\![\boldsymbol{x}_i^\top \boldsymbol{y}_j]\!]_\mathsf{t}$ and $[\![\boldsymbol{y}_j]\!]_2$ for $i = 1, \ldots, q_2$ and $j = 1, \ldots, q_1$, plus the identities. By the SXDH assumption (which we require for our MA-ABE), the $q_1 \cdot q_2$ values $[\![\boldsymbol{x}_i^\top \boldsymbol{y}_j]\!]_\mathsf{t}$ are pseudo-random. The ciphertexts and functional secret keys, which are of total size $(q_1 + q_2) \cdot \mathsf{poly}(\lambda)$ must encode these values of total size $q_1 \cdot q_2 \cdot \mathsf{poly}'(\lambda)$ where $\mathsf{poly}, \mathsf{poly}'$ are polynomials, which is a contradiction. It is not clear how to bypass this impossibility result even in the random oracle model. In fact [AKW18] presents similar impossibility results for FE even in the random oracle model.

Related works. [Kim19] builds a multi-authority ABE for all circuits from LWE for a slightly different notion that the GID model presented here (it can be seen as a relaxation of the GID model). In a recent work, [DKW21a] builds an MA-ABE for DNF formula from LWE, followed by [WWW22] that removed the use of random oracles. In [MJ18], the authors present a decentralized ABE, which is similar to an MA-ABE except the number of authorities of the system is fixed ahead of time, and each authority requires the public keys of the other authorities to generate its share of the user secret key. They realized this notion for the orthogonality-testing predicate (a.k.a. inner-product), which captures NC_0 circuits. Later on, [AYY22] extended their construction to partially hide the predicate in the user secret keys. In the same paper, they also presented a distributed ciphertext-policy ABE for NC_1, based on the LWE assumption and the bilinear generic group model. A distributed ABE is like an MA-ABE except the number of authorities is fixed ahead of time, and the adversary cannot create corrupted authorities with arbitrary public keys, but is instead restricted to (statically) recover the secret keys of honestly generated authorities. In [OT13], the authors build decentralized attribute-based signatures, which generalize the notion of ring signatures, by allowing a user whose attributes satisfy a predicate to sign a message with respect to the predicate. The validity of the signature implies that the signer has valid credentials, but the identity of the signer (or its attributes) remain hidden. As a side result, they also build a multi-authority ABE whose adaptive security is proven under the DLIN assumption in prime-order symmetric pairing groups in the random oracle model. Their scheme supports non-monotone access structures combined with inner-products. However, the security they prove does not handle corruptions of authorities. That is, in the security game, the adversary cannot get the secret key of a set of selected authorities, as is the case for others multi-authority ABE. In a paper concurrent to our work [DKW21b], the authors give the first MA-ABE for monotone span programs from the *search* variant of the Bilinear Diffie Hellman assumption. In their scheme, the size of the MSP, the number of attribute re-use and the size of the attribute universe of each authorities are all a-priori bounded. Their construction also inherits some of the practical deficiencies from prior schemes, namely, it uses symmetric pairings and the ciphertexts contain many target group elements. In [WFL19], the authors build an MA-ABE for bounded collusions (that is, where the number of possible user secret keys that can be corrupted is a priori bounded). Their construction also relies on inner-product FE but which are not identity-based nor handle revocation. They can be built from DDH (without pairing). The main different with our work lies in the unbounded-collusion security feature we achieve, which requires different techniques.

2 Preliminaries

2.1 Notations

We say a function $f : \mathbb{N} \to \mathbb{R}$ is negligible if f is asymptotically dominated by the inverse of any polynomial, i.e for every polynomial $p \in \mathbb{R}[X]$, there exists $\lambda_p \in \mathbb{N}$ such that $|f(\lambda)| \leq |1/p(\lambda)|$ for all $\lambda \geq \lambda_p$. We denote by $|\boldsymbol{v}|$ the length or dimension of vector $\boldsymbol{v}$ and by v_i its i-th component. For any $n \in \mathbb{N}$, we denote $\{1, \ldots, n\}$ by $[n]$. For any column vector $\boldsymbol{u} \in \mathbb{Z}^n$ and $\boldsymbol{v} \in \mathbb{Z}^m$, we denote by $(\boldsymbol{v}, \boldsymbol{u}) \in \mathbb{Z}^{n+m}$ the column vector obtained by concatenating them. Given two matrices (or vectors) $\boldsymbol{A} \in \mathbb{Z}^{m_1 \times n_1}$ and $\boldsymbol{B} \in \mathbb{Z}^{m_2 \times n_2}$, we denote by $\boldsymbol{A} \otimes \boldsymbol{B} \in \mathbb{Z}^{m_1 m_2 \times n_1 n_2}$ their Kronecker product, aka. tensor product defined as follows. For all $i \in [m_1 m_2]$ and $j \in [n_1 n_2]$ which we can write $i = m_1 i_1 + i_2$ with $i_1 \in [m_2]$, $i_2 \in [m_2]$, $j = n_1 j_1 + j_2$ with $j_1 \in [n_2]$, $j_2 \in [n_2]$, the (i, j)'th coordinate of $\boldsymbol{A} \otimes \boldsymbol{B}$ is $a_{i_1,j_1} \cdot b_{i_2,j_2}$.

2.2 Lagrange Interpolation

Let p be a prime and $\mathbb{Z}_p[X]$ denotes the mono-variate polynomials over $\mathbb{Z}_p$. There exists an efficient deterministic algorithm Lagr such that for all $P \in \mathbb{Z}_p[X]$ of degree d, given as input $d+1$ distinct values $x_1, \ldots, x_{d+1} \in \mathbb{Z}_p \setminus \{0\}$, outputs $(\alpha_1, \ldots, \alpha_{d+1}) = \mathsf{Lagr}(x_1, \ldots, \ldots, x_{d+1})$ such that $\alpha_i \in \mathbb{Z}_p$ for all $i \in [d+1]$ and $P(0) = \sum_{i=1}^{d+1} \alpha_i P(x_i)$. The following fact states that when the evaluations of a polynomial P of degree d at only d or less distinct points (different from 0) are given, it is impossible to recover the value $P(0)$, because it is statistically independent from the values at the other points.

Fact 1. Let $d \in \mathbb{N}$, p be a prime, $x_1, \ldots, x_d \in \mathbb{Z}_p \setminus \{0\}$ be d distinct values and P be a uniformly random polynomial over $\mathbb{Z}_p[X]$ of degree d. The value $P(0)$ is statistically independent from $\{P(x_1), \ldots, P(x_d)\}$.

2.3 Access Structure

We recall the definition of monotonic access structures using the language of monotonic span programs [KW93], which capture Boolean formulas. The set of all possible attributes used by an access structure is referred to as the attribute universe. Most of the prior works consider attribute universes of polynomial size (aka small universe) or at least attribute universe of finite size (aka large universe). Here we focus on unbounded attribute universe, where any bit string can serve as an attribute. This is the most advantageous setting in term of flexibility.We denote the set of all possible bit strings by $\{0,1\}^*$.

Definition 1 (Monotonic access structure [Bei96, KW93]). *A* monotonic access structure *is a pair* $(\boldsymbol{M},\rho)$ *where* $\boldsymbol{M} \in \mathbb{Z}_p^{n\times\ell}$ *and* $\rho : [\ell] \to \{0,1\}^*$*. The matrix* $\boldsymbol{M}$ *is used to generate shares as described in Fig. 1, and* ρ *maps each share to its associated attribute. Given a set of attributes* $\mathcal{S} \subseteq \{0,1\}^*$*, we say that*

$$\mathcal{S} \textit{ satisfies } (\boldsymbol{M},\rho) \textit{ iff } \mathbf{1} \in \mathsf{Span}(\boldsymbol{M}_{\mathcal{S}}),$$

where $\mathbf{1} := (1,0,\ldots,0) \in \mathbb{Z}^n$*;* $\boldsymbol{M}_S$ *denotes the collection of vectors* $\{\boldsymbol{M}_j : \rho(j) \in \mathcal{S}\}$ *where* $\boldsymbol{M}_j$ *denotes the* j*'th column of* $\boldsymbol{M}$*; and* Span *refers to linear span of collection of vectors over* $\mathbb{Z}_p$*.*

That is, $\mathcal{S}$ satisfies $(\boldsymbol{M},\rho)$ iff there exists constants $\omega_1,\ldots,\omega_\ell \in \mathbb{Z}_p$ such that

$$\textstyle\sum_{\rho(j)\in S} \omega_j \boldsymbol{M}_j = \mathbf{1} \tag{1}$$

Observe that the constants $\{\omega_i\}$ can be computed in time polynomial in the size of the matrix $\boldsymbol{M}$ via Gaussian elimination.

```
Share($\boldsymbol{M} \in \mathbb{Z}_p^{n\times\ell}, \boldsymbol{a} \in \mathbb{Z}_p^d$):
Sample $\boldsymbol{U} \leftarrow_R \mathbb{Z}_p^{d\times(n-1)}$, and for all $j \in [\ell]$, set $\boldsymbol{a}_j := (\boldsymbol{a}|\boldsymbol{U})\boldsymbol{M}_j \in \mathbb{Z}_p^d$.
Return $\{\boldsymbol{a}_j\}_{j\in[\ell]}$.
```

Fig. 1. Share generation algorithm. Here, $\boldsymbol{M}_j$ denotes the j-th column of $\boldsymbol{M}$. For each $j \in [\ell]$, $\boldsymbol{a}_j$ is a share of the secret $\boldsymbol{a} \in \mathbb{Z}_p^d$.

Now we consider non-monotonic access structures, where ρ maps each share to either an attribute or a *negated* attribute. A set of attribute $\mathcal{S}$ satisfies the non-monotonic access structure $(\boldsymbol{M},\rho)$ if given all the shares that correspond to an attribute in $\mathcal{S}$ or a negated attribute of the form $\neg\mathsf{att}$ where att is not in $\mathcal{S}$, it is possible to recover the secret. For any set $\mathcal{S} \subseteq \{0,1\}^*$, we denote by $\{\neg\}\cdot\mathcal{S}$ the set defined as $\{\neg\mathsf{att}\}_{\mathsf{att}\in\mathcal{S}}$. The formal definition of a non-monotonic access structure is given below.

Definition 2 (Non-monotonic access structure [OSW07]). *A* non-monotonic access structure *is a pair* $(\boldsymbol{M},\rho)$ *where* $\boldsymbol{M} \in \mathbb{Z}_p^{n\times\ell}$ *and* $\rho : [\ell] \to \{0,1\}^* \cup (\{\neg\}\cdot\{0,1\}^*)$*. The matrix* $\boldsymbol{M}$ *is used to generate shares as described in Fig. 1, and* ρ *maps each share to its associated attribute in* $\{0,1\}^*$ *or negated attribute in* $\{\neg\}\cdot\{0,1\}^*$*. Given a set of attributes* $\mathcal{S} \subseteq \{0,1\}^*$*, we say that*

$$\mathcal{S} \textit{ satisfies } (\boldsymbol{M}, \rho) \textit{ iff } \mathbf{1} \in \mathsf{Span}(\boldsymbol{M}_{\mathcal{S}}),$$

where $\mathbf{1} := (1, 0, \ldots, 0) \in \mathbb{Z}^n$*;* $\boldsymbol{M}_S$ *denotes the collection of vectors* $\{\boldsymbol{M}_j : \rho(j) \in \mathcal{S}$ *or* $\rho(j) = \neg\mathsf{att}$ *with* $\mathsf{att} \in \{0,1\}^* \setminus \mathcal{S}\}$*,* $\boldsymbol{M}_j$ *denotes the j'th column of* $\boldsymbol{M}$*, and* Span *refers to the linear span of a collection of (column) vectors over* $\mathbb{Z}_p$*.*

For any set of attributes $\mathcal{S}_{\mathsf{corr}} \subset \{0,1\}^*$*, we say*

$$\mathcal{S} \textit{ satisfies } (\boldsymbol{M}, \rho) \textit{ with corruptions } \mathcal{S}_{\mathsf{corr}} \textit{ iff } \mathbf{1} \in \mathsf{Span}(\boldsymbol{M}_{\mathcal{S},\mathcal{S}_{\mathsf{corr}}}),$$

where $\mathbf{1} := (1, 0, \ldots, 0) \in \mathbb{Z}^n$*;* $\boldsymbol{M}_S$ *denotes the collection of vectors* $\{\boldsymbol{M}_j : \rho(j) \in \mathcal{S} \cup \mathcal{S}_{\mathsf{corr}}$ *or* $\rho(j) = \neg\mathsf{att}$ *with* $\mathsf{att} \in \{0,1\}^* \setminus \mathcal{S}\}$*,* $\boldsymbol{M}_j$ *denotes the j'th column of* $\boldsymbol{M}$*, and* Span *refers to the linear span of a collection of (column) vectors over* $\mathbb{Z}_p$*.*

That is, $\mathcal{S}$ satisfies $(\boldsymbol{M}, \rho)$ iff there exists constants $\omega_1, \ldots, \omega_\ell, \omega'_1, \ldots, \omega'_\ell \in \mathbb{Z}_p$ such that

$$\textstyle\sum_{\rho(j) \in \mathcal{S} \cup \mathcal{S}_{\mathsf{corr}}} \omega_j \boldsymbol{M}_j + \sum_{\rho(j) = \neg\mathsf{att}, \mathsf{att} \notin \mathcal{S}} \omega'_j \boldsymbol{M}_j = \mathbf{1} \qquad (2)$$

Observe that the constants $\{\omega_i, \omega'_i\}$ can be computed in time polynomial in the size of the matrix $\boldsymbol{M}$ via Gaussian elimination. Now we recall a useful fact about access structures represented by span programs.

Lemma 1 ([KW93]). *Let* $(\boldsymbol{M}, \rho)$ *be a non-monotonic access structure where* $\boldsymbol{M} \in \mathbb{Z}_p^{n \times \ell}$*. For all sets* $\mathcal{S}, \mathcal{S}_{\mathsf{corr}} \subseteq \{0,1\}^*$ *such that* $\mathcal{S}$ *does do not satisfy* $(\boldsymbol{M}, \rho)$ *with corruptions* $\mathcal{S}_{\mathsf{corr}}$*, there exists a vector* $\boldsymbol{w}_{\mathcal{S}} \in \mathbb{Z}_p^{\ell-1}$ *such that* $(1, \boldsymbol{w})^\top \boldsymbol{M}_j = 0$ *for all* $j \in [\ell]$ *such that* $\rho(j) \in \mathcal{S} \cup \mathcal{S}_{\mathsf{corr}}$ *or* $\rho(j) = \neg\mathsf{att}$ *with* $\mathsf{att} \in \{0,1\}^* \setminus \mathcal{S}$*.*

2.4 Pairing Groups

Let GGen be a PPT algorithm that on input the security parameter 1^λ, outputs a description $\mathcal{PG} = (p, \mathbb{G}_1, \mathbb{G}_2, P_1, P_2, \mathbb{G}_{\mathsf{t}}, e)$ of pairing groups where $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_{\mathsf{t}}$ are cyclic groups of order p for a 2λ-bit prime p; P_1 and P_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_{\mathsf{t}}$ is an efficiently computable (non-degenerate) bilinear map, thus $P_{\mathsf{t}} := e(P_1, P_2)$ generates $\mathbb{G}_{\mathsf{t}}$.

We use implicit representation of group elements. For $s \in \{1, 2, \mathsf{t}\}$ and $a \in \mathbb{Z}_p$, define $[\![a]\!]_s = a \cdot P_s \in \mathbb{G}_s$ as the implicit representation of a in $\mathbb{G}_s$. More generally, for a matrix $\boldsymbol{A} = (a_{ij}) \in \mathbb{Z}_p^{n \times m}$ we define $[\![\boldsymbol{A}]\!]_s$ as the implicit representation of $\boldsymbol{A}$ in $\mathbb{G}_s$:

$$[\![\boldsymbol{A}]\!]_s := \begin{pmatrix} a_{11} \cdot P_s \ldots a_{1m} \cdot P_s \\ \\ a_{n1} \cdot P_s \ldots a_{nm} \cdot P_s \end{pmatrix} \in \mathbb{G}_s^{n \times m}.$$

Given $[\![a]\!]_1$ and $[\![b]\!]_2$, one can efficiently compute $[\![a \cdot b]\!]_{\mathsf{t}}$ using the pairing e. For matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ of matching dimensions, define $e([\![\boldsymbol{A}]\!]_1, [\![\boldsymbol{B}]\!]_2) := [\![\boldsymbol{AB}]\!]_{\mathsf{t}}$. For any matrix $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{Z}_p^{n \times m}$, any group $s \in \{1, 2, \mathsf{t}\}$, we denote by $[\![\boldsymbol{A}]\!]_s + [\![\boldsymbol{B}]\!]_s = [\![\boldsymbol{A} + \boldsymbol{B}]\!]_s$.

Definition 3 (DDH assumption). *For any adversary $\mathcal{A}$, any group $s \in \{1, 2, \mathsf{t}\}$ and any security parameter λ, let*

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}_s,\mathcal{A}}(\lambda) := |\Pr[1 \leftarrow \mathcal{A}(\mathcal{PG}, [\![\boldsymbol{a}]\!]_s, [\![\boldsymbol{a}r]\!]_s)] - \Pr[1 \leftarrow \mathcal{A}(\mathcal{PG}, [\![\boldsymbol{a}]\!]_s, [\![\boldsymbol{u}]\!]_s)]|,$$

where the probabilities are taken over $\mathcal{PG} \leftarrow_R \mathsf{GGen}(1^\lambda, d)$, $\boldsymbol{a} \leftarrow_R \mathbb{Z}_p^2$, $r \leftarrow_R \mathbb{Z}_p$, $\boldsymbol{u} \leftarrow_R \mathbb{Z}_p^2$, and the random coins of $\mathcal{A}$. We say DDH holds in $\mathbb{G}_s$ if for all PPT adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}_s,\mathcal{A}}(\lambda)$ is a negligible function of λ.

Definition 4 (SXDH assumption). *For any security parameter λ and any pairing group $\mathcal{PG} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, P_1, P_2, e) \leftarrow_R \mathsf{GGen}(1^\lambda)$, we say SXDH holds in $\mathcal{PG}$ if DDH holds in $\mathbb{G}_1$ and $\mathbb{G}_2$.*

It is well known that the DDH and SXDH assumptions are equivalent when the dimensions of the vectors are larger than 2 (for any polynomially large dimensions).

2.5 Functional Encryption

We recall the notion of functional encryption originally given in [BSW11]. Let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of sets, where for each $\lambda \in \mathbb{N}$, $\mathcal{F}_\lambda$ is a set of functions from the message space $\mathcal{X}_\lambda$ to the output space $\mathcal{Y}_\lambda$. A functional encryption scheme for $\mathcal{F}$ consists of the following PPT algorithms.

- $\mathsf{Setup}(1^\lambda) \rightarrow (\mathsf{msk}, \mathsf{pk})$. On input the global parameters gp, it outputs a master secret key msk and a public key pk. The public key is (sometimes implicitly) input to all other algorithms.

- $\mathsf{Enc}(\mathsf{pk}, m) \rightarrow \mathsf{ct}$. On input the public key pk and a message $m \in \mathcal{X}_\lambda$, it outputs a ciphertext ct.

- $\mathsf{KeyGen}(\mathsf{msk}, f) \rightarrow \mathsf{sk}_f$. On input the master secret key msk and a function $f \in \mathcal{F}_\lambda$, it outputs a functional secret key sk_f, which includes the description of the function f.

- $\mathsf{Dec}(\mathsf{pk}, \mathsf{ct}, \mathsf{sk}_f) \rightarrow m$. On input the public key pk, a ciphertext ct and a functional secret key sk_f, the decryption algorithm deterministically outputs a value $\mu \in \mathcal{Y}_\lambda$ (or a special rejection symbol if it fails to decrypt).

Correctness. For all $\lambda \in \mathbb{N}$, all $(\mathsf{pk}, \mathsf{msk})$ in the support of $\mathsf{Setup}(1^\lambda)$, all messages $m \in \mathcal{X}_\lambda$ and all functions $f \in \mathcal{F}_\lambda$, we have

$$\Pr[\mathsf{Dec}(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, m), \mathsf{KeyGen}(\mathsf{msk}, f)) = f(m)] = 1,$$

where the probability is taken over the random coins of Enc and KeyGen.

We now describe the indistinguishability-based security notion for FE.

Adaptive security. Given an FE scheme denoted by FE for $\mathcal{F}$, for any adversary $\mathcal{A}$ and security parameter λ, we define the advantage function:

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{FE}}(\lambda) := \left| \Pr\left[\begin{array}{c} (\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ (m_0, m_1, \mathsf{st}) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{KeyGen}}(\cdot)}(\mathsf{pk}) \\ \beta \leftarrow_R \{0,1\} \\ \mathsf{ct}^\star \leftarrow \mathsf{Enc}(\mathsf{pk}, m_\beta) \\ \beta' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{KeyGen}}(\cdot)}(\mathsf{ct}^\star, \mathsf{st}) \end{array} : \beta' = \beta \right] - \frac{1}{2} \right| ,$$

where the oracle $\mathcal{O}_{\mathsf{KeyGen}}$, when given as input a function $f \in \mathcal{F}_\lambda$, returns $\mathsf{KeyGen}(\mathsf{msk}, f)$ and st denotes the state of the adversary $\mathcal{A}$. We say the adversary $\mathcal{A}$ is admissible if for all functions $f \in \mathcal{F}_\lambda$ queried to $\mathcal{O}_{\mathsf{KeyGen}}$, it holds that $f(m_0) = f(m_1)$. An FE scheme FE is said to be IND-secure if for all PPT admissible adversaries $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{FE}}$ is negligible.

Selective, super-selective security. In the security game above, we say an adversary is selective if it chooses a pair of messages (m_0, m_1) before querying any functional secret key to $\mathcal{O}_{\mathsf{KeyGen}}$. An adversary is said to be super-selective if it is selective and it chooses the queries to $\mathcal{O}_{\mathsf{KeyGen}}$ independently of the challenge ciphertext $\mathsf{ct}^\star$. That is, an FE scheme FE is said to be super-selective if for all admissible PPT adversaries $\mathcal{A}$, the function $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{ssel\text{-}FE}}$ is negligible, where $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{ssel\text{-}FE}}$ is defined for all $\lambda \in \mathbb{N}$ as follows:

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{ssel\text{-}FE}}(\lambda) := \left| \Pr\left[\begin{array}{c} (\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ (m_0, m_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pk}) \\ \mathsf{st}' \leftarrow \mathcal{A}(\mathsf{st})^{\mathcal{O}_{\mathsf{KeyGen}}(\cdot)} \\ \beta \leftarrow_R \{0,1\} \\ \mathsf{ct}^\star \leftarrow \mathsf{Enc}(\mathsf{pk}, m_\beta) \\ \beta' \leftarrow \mathcal{A}(\mathsf{ct}^\star, \mathsf{st}') \end{array} : \beta' = \beta \right] - \frac{1}{2} \right| ,$$

where the oracle $\mathcal{O}_{\mathsf{KeyGen}}$, when given as input a function $f \in \mathcal{F}_\lambda$, returns $\mathsf{KeyGen}(\mathsf{msk}, f)$ and st, st's denote the states of the adversary $\mathcal{A}$. As for the IND-security above, we say the adversary $\mathcal{A}$ is admissible if for all functions $f \in \mathcal{F}_\lambda$ queried to $\mathcal{O}_{\mathsf{KeyGen}}$, it holds that $f(m_0) = f(m_1)$.

2.6 Definition of Multi-Authority ABE

We recall the definition of multi-authority ABE from [LW11]. We assume every authority is identified by a public key. For every authority pk, we denote by $\mathcal{U}_{\mathsf{pk}}$ the associated attribute universe. Without loss of generality, we assume that attribute universes are disjoint for different authorities.

We consider access structures $(\boldsymbol{M}, \rho)$ where $\boldsymbol{M} \in \mathbb{Z}_p^{n \times \ell}$, and ρ maps each row $j \in [\ell]$ to an attribute in $\mathcal{U}_{\theta(j)}$, where θ maps a row $j \in [\ell]$ to the authority who owns the attribute $\rho(j)$. To keep notations simple, we assume the map θ is implicitly part of the description of the access structure.

Definition. A MA-ABE scheme consists of the following PPT algorithms:

- $\mathsf{GlobalSetup}(1^\lambda) \rightarrow \mathsf{gp}$. On input the security parameter, it outputs global parameters, which are input to all other algorithms (usually implicitly).
- $\mathsf{AuthSetup}(\mathsf{gp}) \rightarrow (\mathsf{pk}, \mathsf{sk})$. Each authority runs a setup procedure to generate its own pair of keys. The public key serves as a univocal identifier for the authority, which is associated with an attribute universe denoted by $\mathcal{U}_{\mathsf{pk}}$.
- $\mathsf{Enc}(\boldsymbol{M}, \rho, \mathit{\Pi}) \rightarrow (\mathsf{ct}, \kappa)$. On input an access structure $\boldsymbol{M} \in \mathbb{Z}_p^{n \times \ell}$, $\rho : [\ell] \rightarrow \{0,1\}^*$ and a set of authorities $\mathit{\Pi}$ such that for all columns $j \in [\ell]$, we have $\theta(j) \in \mathit{\Pi}$, the encryption algorithm outputs a ciphertext ct and a symmetric encryption key $\kappa \in \mathcal{K}$. The ciphertext implicitly contains a description of the access structure $(\boldsymbol{M}, \rho)$.
- $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{sk}, \mathsf{gid}, \mathcal{S}) \rightarrow \mathsf{sk}_{\mathsf{gid},\mathcal{S}}$. On input an authority's public key pk and the corresponding secret key sk, a global identifier gid and a set of attribute $\mathcal{S} \subset \mathcal{U}_{\mathsf{pk}}$, the key generation algorithm outputs a user secret key $\mathsf{sk}_{\mathsf{gid},\mathcal{S}}$, which implicitly contains a description of gid and $\mathcal{S}$.
- $\mathsf{Dec}(\mathsf{ct}, \{\mathsf{sk}_{\mathsf{gid},\mathcal{S}_i}\}_i) \rightarrow \kappa/\perp$. On input a ciphertext ct and a set of user secret keys $\{\mathsf{sk}_{\mathsf{gid},\mathcal{S}_i}\}_i$ created for the same global identifier, the decryption algorithm deterministically outputs a symmetric key κ or $\perp$.

Correctness. For all $\lambda \in \mathbb{N}$, all gp in the support of $\mathsf{GlobalSetup}(1^\lambda)$, all $\nu \in \mathbb{N}$, all $(\mathsf{pk}_1, \mathsf{sk}_1), \cdots, (\mathsf{pk}_\nu, \mathsf{sk}_\nu)$ in the support of $\mathsf{Setup}(\mathsf{gp})$, all access structures $(\boldsymbol{M}, \rho)$ associated with the set of authorities $\mathit{\Pi} = \{\mathsf{pk}_1, \ldots, \mathsf{pk}_\nu\}$, all pairs (ct, κ) in the support of $\mathsf{Enc}(\boldsymbol{M}, \rho, \mathit{\Pi})$, all sets of attributes $\mathcal{S}_i \subset \mathcal{U}_{\mathsf{pk}_i}$ for all $i \in [\nu]$ such that $\mathcal{S} = \cup_{i \in [\nu]} \mathcal{S}_i$ satisfies $(\boldsymbol{M}, \rho)$ and all global identifiers $\mathsf{gid} \in \{0,1\}^*$:

$$\Pr\left[\mathsf{Dec}(\mathsf{ct}, \{\mathsf{sk}_{\mathsf{gid},\mathcal{S}_i}\}_{i \in [\nu]}) = \kappa\right] = 1 \ ,$$

where the probability is taken over $\mathsf{sk}_{\mathsf{gid},\mathcal{S}_i} \leftarrow \mathsf{KeyGen}(\mathsf{pk}_i, \mathsf{sk}_i, \mathsf{gid}, \mathcal{S}_i)$ for all $i \in [\nu]$.

Adaptive security. Given a multi-authority ABE denoted by ABE, for any stateful adversary $\mathcal{A}$ and security parameter λ, we define the advantage function:

$$\mathsf{Adv}^{\mathsf{ABE}}_{\mathcal{A}}(\lambda) :=$$
$$\left| \Pr\left[\begin{array}{c} \mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda) \\ (\boldsymbol{M}, \rho, \mathit{\Pi}_{\mathsf{hon}}, \mathit{\Pi}_{\mathsf{corr}}) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{create}}, \mathcal{O}_{\mathsf{corr}}(\cdot), \mathcal{O}_{\mathsf{KeyGen}}(\cdot,\cdot,\cdot)}(\mathsf{gp}) \\ (\mathsf{ct}^\star, \kappa) \leftarrow \mathsf{Enc}(\boldsymbol{M}, \rho, \mathit{\Pi}) \\ \beta \leftarrow_R \{0,1\};\ K_0 := \kappa;\ K_1 \leftarrow_R \mathcal{K} \\ \beta' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{corr}}(\cdot), \mathcal{O}_{\mathsf{KeyGen}}(\cdot,\cdot,\cdot)}(\mathsf{ct}^\star, K_\beta) \end{array} : \beta' = \beta \right] - \frac{1}{2} \right| .$$

The oracles are defined as follows:

- $\mathcal{O}_{\mathsf{create}}$: runs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{AuthSetup}(\mathsf{gp})$, adds pk to the sets of honest authorities denoted by $\mathcal{S}_{\mathsf{hon}}$ (initially empty) and returns pk.
- $\mathcal{O}_{\mathsf{corr}}(\mathsf{pk})$: if $\mathsf{pk} \in \mathcal{S}_{\mathsf{hon}}$, it returns the associated secret key sk and removes pk from $\mathcal{S}_{\mathsf{hon}}$.
- $\mathcal{O}_{\mathsf{KeyGen}}(\mathsf{pk}, \mathsf{gid}, \mathcal{S})$: if $\mathsf{pk} \in \mathcal{S}_{\mathsf{hon}}$ and $\mathcal{S} \subset \mathcal{U}_{\mathsf{pk}}$, it returns $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{sk}, \mathsf{gid}, \mathcal{S})$ where sk is the secret key associated with pk; otherwise, it returns $\perp$. This oracle can be queried at most once per $(\mathsf{pk}, \mathsf{gid})$ pair. That is, there cannot be two queries of the form $(\mathsf{pk}, \mathsf{gid}, \mathcal{S})$ and $(\mathsf{pk}, \mathsf{gid}, \mathcal{S}')$ for different $\mathcal{S} \neq \mathcal{S}'$ to $\mathcal{O}_{\mathsf{KeyGen}}$. This restriction is necessary for non-monotonic access structure (see Remark 2).

The adversary $\mathcal{A}$ outputs an access structure $(\boldsymbol{M}, \rho)$ with respect to the authorities $\mathit{\Pi} = \mathit{\Pi}_{\mathsf{hon}} \cup \mathit{\Pi}_{\mathsf{corr}}$, where $\mathit{\Pi}_{\mathsf{hon}}$ denotes the set of honest authorities, that is, which have been created via $\mathcal{O}_{\mathsf{create}}$, and which have not been queried to $\mathcal{O}_{\mathsf{corr}}$ (they can still be queried to $\mathcal{O}_{\mathsf{corr}}$ later on), whereas $\mathit{\Pi}_{\mathsf{corr}}$ denotes the set of corrupted authorities, that is, authorities created via $\mathcal{O}_{\mathsf{create}}$ that have been subsequently queried to $\mathcal{O}_{\mathsf{corr}}$, or authorities whose public keys were maliciously created by the adversary $\mathcal{A}$ himself. We require that $\mathit{\Pi}_{\mathsf{corr}}$ contains not only the public keys of the corrupted authorities, but also their associated secret keys[3].

We denote by $\mathcal{Q}_{\mathsf{KeyGen}}$ the set of queries to $\mathcal{O}_{\mathsf{KeyGen}}$, $\mathcal{S}_{\mathsf{hon}} \subseteq \mathit{\Pi}_{\mathsf{hon}}$ the set of authorities in $\mathit{\Pi}_{\mathsf{hon}}$ that are still honest at the end of the experiment, $\mathcal{S}_{\mathsf{corr}} = \mathit{\Pi}_{\mathsf{corr}} \cup \mathit{\Pi}_{\mathsf{hon}} \backslash \mathcal{S}_{\mathsf{hon}}$, $\mathit{\Sigma}_{\mathsf{corr}} = \cup_{\mathsf{pk} \in \mathcal{S}_{\mathsf{corr}}} \mathcal{U}_{\mathsf{pk}}$, and for every global identifier $\mathsf{gid} \in \{0,1\}^*$, $\mathcal{S}_{\mathsf{gid}} = \cup_{\mathsf{pk} \in \mathcal{S}_{\mathsf{hon}}, (\mathsf{pk}, \mathsf{gid}, \mathcal{S}) \in \mathcal{Q}_{\mathsf{KeyGen}}} \mathcal{S}$. We say the adversary $\mathcal{A}$ is admissible if for all $\mathsf{gid} \in \{0,1\}^*$, $\mathcal{S}_{\mathsf{gid}}$ does not satisfy $(\boldsymbol{M}, \rho)$ with corruptions $\mathit{\Sigma}_{\mathsf{corr}}$ (as per Definition 1). We say ABE is adaptively secure if for all PPT admissible adversaries $\mathcal{A}$, there exists a negligible function ν such that for all $\lambda \in \mathbb{N}$, $\mathsf{Adv}^{\mathsf{ABE}}_{\mathcal{A}}(\lambda) \leq \nu(\lambda)$.

Static corruptions. We say an ABE is secure with static corruptions if the adversary does not have access to the oracle $\mathcal{O}_{\mathsf{corr}}$. He can still create authorities

[3] The restriction which requires that the adversary provide the secret keys of the corrupted authorities in $\mathit{\Pi}_{\mathsf{corr}}$ can be lifted via a generic use of Zero-Knowledge Argument of Knowledge. See Remark 3 for further details.

maliciously as part of $\mathit{\Pi}_{\mathsf{corr}}$, but all authorities created by $\mathcal{O}_{\mathsf{create}}$ remain honest throughout the experiment.

Selective, super-selective security. In the security game above, we say an adversary is selective if it chooses the tuple $(\boldsymbol{M}, \rho, \mathit{\Pi}_{\mathsf{corr}}, \mathit{\Pi}_{\mathsf{hon}})$ before querying any user secret key to $\mathcal{O}_{\mathsf{KeyGen}}$. An adversary is said to be super-selective if it is selective and it chooses the queries to $\mathcal{O}_{\mathsf{KeyGen}}$ independently of the challenge ciphertext $\mathsf{ct}^{\star}$. That is, an MA-ABE scheme ABE is said to be super-selective if for all admissible PPT adversaries $\mathcal{A}$, the function $\mathsf{Adv}^{\mathsf{ssel\text{-}ABE}}_{\mathcal{A}}$ is negligible, where $\mathsf{Adv}^{\mathsf{ssel\text{-}ABE}}_{\mathcal{A}}$ is defined for all $\lambda \in \mathbb{N}$ as follows:

$$\mathsf{Adv}^{\mathsf{ABE}}_{\mathcal{A}}(\lambda) :=$$

$$\left| \Pr\left[\begin{array}{r} \mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^{\lambda}) \\ (\boldsymbol{M}, \rho, \mathit{\Pi}_{\mathsf{hon}}, \mathit{\Pi}_{\mathsf{corr}}, \mathsf{st}) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{create}}, \mathcal{O}_{\mathsf{corr}}(\cdot)}(\mathsf{gp}) \\ \mathsf{st}' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{corr}}(\cdot), \mathcal{O}_{\mathsf{KeyGen}}(\cdot,\cdot,\cdot)}(\mathsf{st}) \\ (\mathsf{ct}^{\star}, \kappa) \leftarrow \mathsf{Enc}(\boldsymbol{M}, \rho, \mathit{\Pi}) \\ \beta \leftarrow_R \{0,1\};\ K_0 := \kappa;\ K_1 \leftarrow_R \mathcal{K} \\ \beta' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{corr}}(\cdot)}(\mathsf{ct}^{\star}, K_{\beta}, \mathsf{st}') \end{array} : \beta' = \beta \right] - \frac{1}{2} \right| .$$

where the oracles are defined as above, and st, st' denote the states of the adversary $\mathcal{A}$.

Remark 2 (At most one user secret key query per gid*).* In the definitions above, we restrict the adversary to query the oracle $\mathcal{O}_{\mathsf{KeyGen}}$ at most once per $(\mathsf{pk}, \mathsf{gid})$ pair . This restriction is necessary when considering non-monotone access structure. In fact, security relies on the fact that users only obtain user secret keys associated to the set of *all* attributes they possess. Giving the adversary access to at most one query to $\mathcal{O}_{\mathsf{KeyGen}}$ per $(\mathsf{pk}, \mathsf{gid})$ is one way to ensure this is the case.

For instance, suppose a user Alice possesses the attributes att_1 and att_2 that are owned by an authority. Alice should not be able to obtain user secret keys associated to strict subsets of $\{\mathsf{att}_1, \mathsf{att}_2\}$. If for example she obtains a user secret key for $\{\mathsf{att}_1\}$, she would be able to decrypt a ciphertext associated with an access structure excluding users possessing att_2.

Remark 3 (Stronger security via ZK-AoK). In the security definition above, we require the adversary to provide not only the public keys, but also the secret keys of all the authorities in $\mathit{\Pi}_{\mathsf{corr}}$. It is possible to lift this restriction, and thereby strengthen the security definition, using standard techniques involving Zero-Knowledge Argument of Knowledge (ZK-AoK). Any authority must publish not only a public key, but also an argument of knowledge of the associated secret key. The zero-knowledge property ensures that nothing is revealed about the secret key, and the argument of knowledge property forces the issuer to know the associated secret key. This way, the adversary must know the secret key associated to any authority it creates maliciously, since it has to provide an argument of knowledge. Note that in our ABE constructions we use a ZK-AoK for a very simple language that admits an efficient sigma protocol, that can be made non-interactive with the Fiat-Shamir heuristic. Consequently, strengthening the security comes at a modest efficiency cost. In the rest of this paper, we focus on the weaker security definition, which is easier to prove.

3 Inner-Product FE

3.1 Identity-Based Inner-Product FE

We recall the definition of Identity-Based Inner-Product Functional Encryption (ID-IPFE) which is a particular case of Functional Encryption where the family $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ is as follows. Let d be a polynomial and GGen a pairing group generator. For every $\lambda \in \mathbb{N}$, the set of functions $\mathcal{F}_\lambda$ is associated with a pairing group $(p, \mathbb{G}_1, \mathbb{G}_2, P_1, P_2, \mathbb{G}_\mathsf{t}, e) = \mathsf{GGen}(1^\lambda)$, where p is a prime which denotes the order of the groups $\mathbb{G}_1, \mathbb{G}_2$, and $\mathbb{G}_\mathsf{t}$. We assume the pairing group $\mathcal{PG}$ is given as input of the setup algorithm. The message space $\mathcal{X}_\lambda = \mathbb{Z}_p^{d(\lambda)} \times \mathbb{Z}_p$. That is, every message is of the form $(\boldsymbol{x}, \mathsf{id})$, where $\boldsymbol{x} \in \mathbb{Z}_p^{d(\lambda)}$ is referred to as the message vector, and $\mathsf{id} \in \mathbb{Z}_p$ is referred to as the identity. The function space $\mathcal{F}_\lambda = \mathbb{G}_2^{d(\lambda)} \times \mathbb{Z}_p$. Every function is of the form $([\![\boldsymbol{y}]\!]_2, \mathsf{id}')$ where $[\![\boldsymbol{y}]\!]_2 \in \mathbb{G}_2^{d(\lambda)}$ and $\mathsf{id}' \in \mathbb{Z}_p$. Decryption recovers the inner product $[\![\boldsymbol{x}^\top \boldsymbol{y}]\!]_\mathsf{t} \in \mathbb{G}_\mathsf{t}$ when $\mathsf{id} = \mathsf{id}'$. When $\mathsf{id}' \neq \mathsf{id}$, the vector $\boldsymbol{x}$ remains hidden. In both cases, the vector $[\![\boldsymbol{y}]\!]_2$ and the identities id and id' are revealed.

In [DP19, TT18], the authors give an unbounded variant of the related family where functions are of the form $(\boldsymbol{y}, \mathsf{id}) \in \mathbb{Z}_p^{d(\lambda)} \times \mathbb{Z}_p$, that is, the vector $\boldsymbol{y}$ needs to be known in $\mathbb{Z}_p^{d(\lambda)}$ instead of $\mathbb{G}_2^{d(\lambda)}$. In our MA-ABE that uses the ID-IPFE as a building block, the party generating the functional secret keys only know the value $[\![\boldsymbol{y}]\!]_2 \in \mathbb{G}_2^{d(\lambda)}$, which prevents us from using their scheme. In [ACGU20], the authors present an ID-IPFE for the functions described above (where $\mathcal{F}_\lambda = \mathbb{G}_2^{d(\lambda)} \times \mathbb{Z}_p$) which is selectively secure under the SXDH assumption. They also present an adaptively secure construction but only for the messages $(\boldsymbol{x}, \mathsf{id})$ and functions $([\![\boldsymbol{y}]\!]_2, \mathsf{id}')$ such that $\boldsymbol{x}^\top \boldsymbol{y}$ is small (i.e. lies in a set of polynomial size), which is not the case for our application. Indeed the value of the inner product $[\![\boldsymbol{x}^\top \boldsymbol{y}]\!]_\mathsf{t}$ in our case will be well-spread in the full group $\mathbb{G}_\mathsf{t}$. This prevents from using the adaptively secure scheme from [ACGU20]. It is an open problem to build an adaptively secure ID-IPFE for large values.

3.2 Inner-Product FE with Revocations

Here we consider a Functional Encryption scheme for the family $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ where for all $\lambda \in \mathbb{N}$, $\mathcal{X}_\lambda = \mathbb{Z}_p^{d(\lambda)} \times \mathbb{Z}_p$, $\mathcal{F}_\lambda = \mathbb{G}_2^{d(\lambda)} \times \mathcal{S}_t$, $\mathcal{S}_t$ denotes all the sets of size t included in $\mathbb{Z}_p$, and p is a prime which denotes the order of a pairing group $\mathcal{PG} = (p, \mathbb{G}_1, \mathbb{G}_2, P_1, P_2, \mathbb{G}_\mathsf{t}, e)$. We assume the pairing group $\mathcal{PG}$ is given as input of the setup algorithm. For every message of the form $(\boldsymbol{x}, \mathsf{id})$ where $\boldsymbol{x} \in \mathbb{Z}_p^{d(\lambda)}$ and $\mathsf{id} \in \mathbb{Z}_p$, and every function of the form $(\llbracket \boldsymbol{y} \rrbracket_2, \mathcal{S})$ where $\mathcal{S} \subset \mathbb{Z}_p$ is of size t, decryption recovers $\llbracket \boldsymbol{x}^\top \boldsymbol{y} \rrbracket_\mathsf{t}$ when $\mathsf{id} \notin \mathcal{S}$. When $\mathsf{id} \in \mathcal{S}$, then the vector $\boldsymbol{x}$ remains hidden. In both cases, the identity id, the set $\mathcal{S}$ and the vector $\llbracket \boldsymbol{y} \rrbracket_2$ are revealed. Note that the set $\mathcal{S}$ associated to each functional secret key is required to be of size *exactly* t. We argue in Section 3.3 how to remove this restriction and have sets of size at most t. We now give the first construction of such an FE scheme, whose selective security we prove under SXDH. It is described in Fig. 2. It makes use of Lagrange interpolation, described in Section 2.2.

$\underline{\mathsf{Setup}(1^\lambda, \mathcal{PG}):}$

Given as input the security parameter $\lambda \in \mathbb{N}$ and a pairing group $\mathcal{PG} := (p, \mathbb{G}_1, \mathbb{G}_2, P_1, P_2, \mathbb{G}_\mathsf{t}, e)$, it samples $\boldsymbol{U}_0, \dots, \boldsymbol{U}_t \leftarrow_R \mathbb{Z}_p^{2\times 2}$, $\boldsymbol{a}, \boldsymbol{b} \leftarrow_R \mathbb{Z}_p^2$, $\boldsymbol{V} \leftarrow_R \mathbb{Z}_p^{d\times 2}$.

For all $x \in \mathbb{Z}_p$, we define $\boldsymbol{P}(x) = \boldsymbol{U}_0 + \boldsymbol{U}_1 x + \cdots + \boldsymbol{U}_t x^t \in \mathbb{Z}_p^{2\times 2}$.

Set $\mathsf{msk} = \Big(\boldsymbol{V}, \llbracket \boldsymbol{b} \rrbracket_2, (\llbracket \boldsymbol{U}_i^\top \boldsymbol{b} \rrbracket_2)_{i\in\{0,\dots,t\}}\Big)$, $\mathsf{pk} = \Big(\llbracket \boldsymbol{a} \rrbracket_1, (\llbracket \boldsymbol{U}_i \boldsymbol{a} \rrbracket_1)_{i\in\{0,\dots,t\}}, \llbracket \boldsymbol{V}\boldsymbol{a} \rrbracket_1\Big)$ and output $(\mathsf{msk}, \mathsf{pk})$.

$\underline{\mathsf{Enc}(\mathsf{pk}, \boldsymbol{x}, \mathsf{id}):}$

Given pk, $\boldsymbol{x} \in \mathbb{Z}_p^d$, $\mathsf{id} \in \mathbb{Z}_p$, it samples $r \leftarrow_R \mathbb{Z}_p$ and returns $\mathsf{ct} = \big(\llbracket \boldsymbol{a}r \rrbracket_1, \llbracket \boldsymbol{x} + \boldsymbol{V}\boldsymbol{a}r \rrbracket_1, \llbracket \boldsymbol{P}(\mathsf{id})\boldsymbol{a}r \rrbracket_1\big) \in \mathbb{G}_1^{2+d+2}$.

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \llbracket \boldsymbol{y} \rrbracket_2, \mathcal{S}):}$

Given msk, $\llbracket \boldsymbol{y} \rrbracket_2 \in \mathbb{G}_2^d$ and a set $\mathcal{S} \subset \mathbb{Z}_p$ of size t, it samples $s \leftarrow_R \mathbb{Z}_p$,
and returns $\mathsf{sk} = \Big(\llbracket \boldsymbol{b}s \rrbracket_2, \llbracket \boldsymbol{V}^\top \boldsymbol{y} + \boldsymbol{P}(0)^\top \boldsymbol{b}s \rrbracket_2, (\llbracket \boldsymbol{P}(\mathsf{id}_j)^\top \boldsymbol{b}s \rrbracket_2)_{\mathsf{id}_j \in \mathcal{S}}\Big) \in \mathbb{G}_2^{2+2+2t}$.

$\underline{\mathsf{Dec}(\mathsf{ct}, \mathsf{id}, \mathsf{sk}, \llbracket \boldsymbol{y} \rrbracket_2, \mathcal{S}):}$

Parse ct as $(\llbracket \boldsymbol{c}_1 \rrbracket_1, \llbracket \boldsymbol{c}_2 \rrbracket_1, \llbracket \boldsymbol{c}_3 \rrbracket_1) \in \mathbb{G}_1^2 \times \mathbb{G}_1^d \times \mathbb{G}_1^2$, sk as $\big(\llbracket \boldsymbol{k}_1 \rrbracket_2, \llbracket \boldsymbol{k}_2 \rrbracket_2, (\llbracket \boldsymbol{k}_{j,3} \rrbracket_2)_{j\in[t]}\big) \in \mathbb{G}_2^2 \times \mathbb{G}_2^2 \times \big(\mathbb{G}_1^2\big)^t$, and $\mathcal{S}$ as $\mathcal{S} = \{\mathsf{id}_1, \dots, \mathsf{id}_t\}$.

For all $j \in [t]$, compute $\llbracket \gamma_j \rrbracket_\mathsf{t} = e(\llbracket \boldsymbol{c}_1^\top \rrbracket_1, \llbracket \boldsymbol{k}_{j,3} \rrbracket_2)$ and $\llbracket \gamma_{t+1} \rrbracket_\mathsf{t} = e(\llbracket \boldsymbol{c}_3^\top \rrbracket_1, \llbracket \boldsymbol{k}_1 \rrbracket_2)$.

Compute $(\alpha_1, \dots, \alpha_{t+1}) = \mathsf{Lagr}(\mathsf{id}_1, \dots, \mathsf{id}_t, \mathsf{id})$. Return $e(\llbracket \boldsymbol{c}_2^\top \rrbracket_1, \llbracket \boldsymbol{y} \rrbracket_2) \cdot \prod_{j\in[t+1]} \llbracket \gamma_j \rrbracket_\mathsf{t}^{\alpha_j} / e(\llbracket c_1 \rrbracket_1, \llbracket k_2 \rrbracket_2)$.

Fig. 2. Inner-product FE with revocations for d-dimensional vectors and sets of size t. Its selective security is proven under SXDH. The algorithm Lagr is described in Section 2.2.

Correctness. Since $\mathsf{id} \notin \mathcal{S}$, we can use the correctness of the algorithm Lagr, which states that: $\prod_{j\in[t+1]} \llbracket \gamma_j \rrbracket_\mathsf{t}^{\alpha_j} = \llbracket s\boldsymbol{b}^\top \boldsymbol{P}(0)\boldsymbol{a}r \rrbracket_\mathsf{t}$. Thus, the decryption computes:

$$\begin{aligned} & e(\llbracket \boldsymbol{c}_2^\top \rrbracket_1, \llbracket \boldsymbol{y} \rrbracket_2) \cdot \prod_{j\in[t+1]} \llbracket \gamma_j \rrbracket_{\mathsf{t}}^{\alpha_j} \,/\, e(\llbracket c_1 \rrbracket_1, \llbracket k_2 \rrbracket_2) \\ &= \left\llbracket (\boldsymbol{x}+\boldsymbol{V}\boldsymbol{a}r)^\top \boldsymbol{y} + s\boldsymbol{b}^\top \boldsymbol{P}(0)\boldsymbol{a}r - r\boldsymbol{a}^\top(\boldsymbol{V}^\top\boldsymbol{y}+\boldsymbol{P}(0)^\top\boldsymbol{b}s)\right\rrbracket_{\mathsf{t}} \\ &= \llbracket \boldsymbol{x}^\top\boldsymbol{y}\rrbracket_{\mathsf{t}} \,. \end{aligned}$$

Theorem 1 (Selective security). *The scheme presented in Fig. 2 is selectively secure under the SXDH assumption.*

Proof. We proceed via a series of hybrid games described bellow (the differences from one game to the next are highlighted in red).

$\underline{\mathsf{Game}_0\text{:}}$ is the game from the selective security definition in Section 2.5. Recall that the adversary $\mathcal{A}$ first receives $\mathsf{pk} = \Big(\llbracket \boldsymbol{a} \rrbracket_1, (\llbracket \boldsymbol{U}_i\boldsymbol{a} \rrbracket_1)_{i\in\{0,\ldots,t\}}, \llbracket \boldsymbol{V}\boldsymbol{a} \rrbracket_1\Big)$. Then, it chooses a pair of messages $((\boldsymbol{x}_0, \mathsf{id}_0), (\boldsymbol{x}_1, \mathsf{id}_1))$, upon which it receives $\mathsf{ct}^\star = (\llbracket \boldsymbol{a}r \rrbracket_1, \llbracket \boldsymbol{x}_\beta + \boldsymbol{V}\boldsymbol{a}r \rrbracket_1, \llbracket \boldsymbol{P}(\mathsf{id}_\beta)\boldsymbol{a}r \rrbracket_1)$, where $\beta \leftarrow_R \{0,1\}$. Afterwards, it can query its oracle $\mathcal{O}_{\mathsf{KeyGen}}$ on inputs of the form $(\llbracket \boldsymbol{y} \rrbracket_2, \mathcal{S})$, upon which it gets $\mathsf{sk} = (\llbracket \boldsymbol{b}s \rrbracket_2, \llbracket \boldsymbol{V}^\top\boldsymbol{y} + \boldsymbol{P}(0)^\top\boldsymbol{b}s \rrbracket_2, (\llbracket \boldsymbol{P}(\mathsf{id}_j)^\top \boldsymbol{b}s \rrbracket_2)_{\mathsf{id}_j\in\mathcal{S}})$. The adversary $\mathcal{A}$ is admissible, which means that $\mathsf{id}_0 = \mathsf{id}_1$, which we denote by $\mathsf{id}^\star = \mathsf{id}_0 = \mathsf{id}_1$, and that for all queries $(\llbracket \boldsymbol{y} \rrbracket_2, \mathcal{S})$ to $\mathcal{O}_{\mathsf{KeyGen}}$, we have $\mathsf{id}^\star \in \mathcal{S}$ or ($\mathsf{id}^\star \notin \mathcal{S}$ and $\boldsymbol{x}_0^\top\boldsymbol{y} = \boldsymbol{x}_1^\top\boldsymbol{y}$). At the end, the adversary $\mathcal{A}$ outputs a guess β'.

$\underline{\mathsf{Game}_1\text{:}}$ we change the way the challenge ciphertext is computed. Namely, we have now

$$\mathsf{ct}^\star = \big(\llbracket \boldsymbol{z} \rrbracket_1, \llbracket \boldsymbol{x}_\beta + \boldsymbol{V}\boldsymbol{z} \rrbracket_1, \llbracket \boldsymbol{P}(\mathsf{id}^\star)\boldsymbol{z} \rrbracket_1\big) \ ,$$

where $\boldsymbol{z} \leftarrow_R \mathbb{Z}_p^2$. We prove that $\mathsf{Game}_0 \approx_c \mathsf{Game}_1$ by the DDH assumption in $\mathbb{G}_1$. Namely, we have $(\llbracket \boldsymbol{a} \rrbracket_1, \llbracket \boldsymbol{a}r \rrbracket_1) \approx_c (\llbracket \boldsymbol{a} \rrbracket_1, \llbracket \boldsymbol{z} \rrbracket_1)$ where the leftmost distribution corresponds to Game_0, whereas the rightmost distribution corresponds to Game_1.

$\underline{\mathsf{Game}_2\text{:}}$ we change the way the challenge ciphertext is computed. Namely, we have now

$$\mathsf{ct}^\star = \big(\llbracket \boldsymbol{z} \rrbracket_1, \llbracket \boldsymbol{x}_\beta + \boldsymbol{V}\boldsymbol{z} \rrbracket_1, \llbracket \boldsymbol{P}(\mathsf{id}^\star)\boldsymbol{z} \rrbracket_1\big) \ ,$$

where $\boldsymbol{z} \leftarrow_R \mathbb{Z}_p^2 \setminus \mathsf{Span}(\boldsymbol{a})$. Here $\mathsf{Span}(\boldsymbol{a})$ denotes the set of vectors proportional to $\boldsymbol{a}$. The cardinal of $\mathsf{Span}(\boldsymbol{a})$ is p, thus, the statistical distance between the uniform distribution over $\mathbb{Z}_p^2 \setminus \mathsf{Span}(\boldsymbol{a})$ and uniform over $\mathbb{Z}_p^2$ is $1/p$, and $\mathsf{Game}_1 \approx_s \mathsf{Game}_2$.

$\underline{\mathsf{Game}_3\text{:}}$ we change the way the functional keys and the challenge ciphertext are computed. Namely, the ciphertext is now of the form:

$$\mathsf{ct}^\star = \big(\llbracket \boldsymbol{z} \rrbracket_1, \llbracket \boldsymbol{V}\boldsymbol{z} \rrbracket_1, \llbracket \boldsymbol{P}(\mathsf{id}^\star)\boldsymbol{z} \rrbracket_1\big) \ .$$

Note that the ciphertext does not depend on the message $\boldsymbol{x}_\beta$ anymore. Each query $(\llbracket \boldsymbol{y} \rrbracket_2, \mathcal{S})$ to $\mathcal{O}_{\mathsf{KeyGen}}$ is now answered with

$$\Big(\llbracket \boldsymbol{b}s \rrbracket_2, \llbracket \boldsymbol{V}^\top\boldsymbol{y} - \boldsymbol{a}^\perp \cdot \boldsymbol{x}_\beta^\top\boldsymbol{y} + \boldsymbol{P}(0)^\top\boldsymbol{b}s \rrbracket_2, (\llbracket \boldsymbol{P}(\mathsf{id}_j)^\top \boldsymbol{b}s \rrbracket_2)_{\mathsf{id}_j\in\mathcal{S}}\Big) \ ,$$

where $\boldsymbol{a}^\perp \in \mathbb{Z}_p^2$ is the vector such that $\boldsymbol{a}^\top \boldsymbol{a}^\perp = 0$ and $\boldsymbol{z}^\top \boldsymbol{a}^\perp = 1$. Game_2 and Game_3 are identically distributed, since for all $\boldsymbol{x}_\beta \in \mathbb{Z}_p^d$, all $\boldsymbol{a}^\perp \in \mathbb{Z}_p^2$, the following are identically distributed: $\{\boldsymbol{V} \leftarrow_R \mathbb{Z}_p^{d\times 2} : \boldsymbol{V}\}$ and $\{\boldsymbol{V} \leftarrow_R \mathbb{Z}_p^{d\times 2} : \boldsymbol{V} - \boldsymbol{x}_\beta(\boldsymbol{a}^\perp)^\top\}$. The former distribution corresponds to Game_2 with some pre and post-processing, whereas the latter corresponds to Game_3 with the same pre and post-processing. Note that Game_3 crucially relies on the fact that the adversary is selective, since the vector $\boldsymbol{x}_\beta$ needs to be known to generate all functional secret keys.

$\underline{\mathsf{Game}_4}$: we change the way the functional keys are computed. Namely, each query $(\llbracket \boldsymbol{y} \rrbracket_2, \mathcal{S})$ to $\mathcal{O}_{\mathsf{KeyGen}}$ is now answered with

$$\Big(\llbracket \boldsymbol{b}s \rrbracket_2, \llbracket \boldsymbol{V}^\top \boldsymbol{y} - \boldsymbol{1}_{\mathsf{id}^\star \notin \mathcal{S}} \boldsymbol{a}^\perp \boldsymbol{x}_\beta^\top \boldsymbol{y} + \boldsymbol{P}(0)^\top \boldsymbol{b}s \rrbracket_2, (\llbracket \boldsymbol{P}(\mathsf{id}_j)^\top \boldsymbol{b}s \rrbracket_2)_{\mathsf{id}_j \in \mathcal{S}}\Big) .$$

That is, now we only have the term $\boldsymbol{a}^\perp \boldsymbol{x}_\beta^\top \boldsymbol{y}$ for functional key queries $(\boldsymbol{y}, \mathcal{S})$ where $\mathsf{id}^\star \notin \mathcal{S}$. To transition from Game_3 to Game_4, we use the following hybrid games.

$\underline{\mathsf{Game}_{3.i}}$: for all $i \in \{0, \ldots, Q\}$, where Q denotes the number of functional key queries, $\mathsf{Game}_{3.i}$ is defined as Game_4 for the first i'th key queries and as Game_3 for the last $Q - i$ queries. By definition we have $\mathsf{Game}_3 = \mathsf{Game}_{3.0}$ and $\mathsf{Game}_4 = \mathsf{Game}_{3.Q}$. It suffices to show that for all $i \in [Q]$, $\mathsf{Game}_{3.i-1} \approx_c \mathsf{Game}_{3.i}$. To do so, we introduce new intermediate games, defined as follows.

$\underline{\mathsf{Game}_{3.i-1.1}}$: is defined as $\mathsf{Game}_{3.i-1}$, except the i'th query to $\mathcal{O}_{\mathsf{KeyGen}}$, denoted by $(\llbracket \boldsymbol{y}_i \rrbracket_2, \mathcal{S}_i)$, is now answered with

$$\Big(\llbracket \boldsymbol{d} \rrbracket_2, \llbracket \boldsymbol{V}^\top \boldsymbol{y}_i - \boldsymbol{a}^\perp \cdot \boldsymbol{x}_\beta^\top \boldsymbol{y}_i + \boldsymbol{P}(0)^\top \boldsymbol{d} \rrbracket_2, (\llbracket \boldsymbol{P}(\mathsf{id}_j)^\top \boldsymbol{d} \rrbracket_2)_{\mathsf{id}_j \in \mathcal{S}_i}\Big) ,$$

where $\boldsymbol{d} \leftarrow_R \mathbb{Z}_p^2$. We have $\mathsf{Game}_{3.i-1} \approx_c \mathsf{Game}_{3.i-1.1}$ by the DDH assumption in $\mathbb{G}_2$, which states that $(\llbracket \boldsymbol{b} \rrbracket_2, \llbracket \boldsymbol{b}s_i \rrbracket_2) \approx_c (\llbracket \boldsymbol{b} \rrbracket_2, \llbracket \boldsymbol{d} \rrbracket_2)$ where $\boldsymbol{b}, \boldsymbol{d} \leftarrow_R \mathbb{Z}_p^2, s_i \leftarrow_R \mathbb{Z}_p$. The former distribution corresponds to $\mathsf{Game}_{3.i-1}$ with some efficient post-processing, whereas the latter corresponds to $\mathsf{Game}_{3.i-1.1}$ with the same post-processing.

$\underline{\mathsf{Game}_{3.i-1.2}}$: is defined as $\mathsf{Game}_{3.i-1.1}$, except the vector $\boldsymbol{d}$ used to compute the i'th queried functional secret key is sampled as $\boldsymbol{d} \leftarrow_R \mathbb{Z}_p^2 \setminus \mathsf{Span}(\boldsymbol{b})$, instead of uniformly random over $\mathbb{Z}_p^2$. Since the cardinal of $\mathsf{Span}(\boldsymbol{b})$ is at most p, the uniform distribution over $\mathbb{Z}_p^2 \setminus \mathsf{Span}(\boldsymbol{b})$ has statistical distance at most $1/p$ with the uniform distribution over $\mathbb{Z}_p^2$. Thus, $\mathsf{Game}_{3.i-1.1} \approx_s \mathsf{Game}_{3.i-1.2}$.

$\underline{\mathsf{Game}_{3.i-1.3}}$: is defined as $\mathsf{Game}_{3.i-1.2}$, except the i'th query to $\mathcal{O}_{\mathsf{KeyGen}}$ is now answered with

$$\Big(\llbracket \boldsymbol{d} \rrbracket_2, \llbracket \boldsymbol{V}^\top \boldsymbol{y}_i - \boldsymbol{1}_{\mathsf{id}^\star \notin \mathcal{S}_i} \boldsymbol{a}^\perp \boldsymbol{x}_\beta^\top \boldsymbol{y}_i + \boldsymbol{P}(0)^\top \boldsymbol{d} \rrbracket_2, (\llbracket \boldsymbol{P}(\mathsf{id}_j)^\top \boldsymbol{d} \rrbracket_2)_{\mathsf{id}_j \in \mathcal{S}_i}\Big) ,$$

where $\boldsymbol{d} \leftarrow_R \mathbb{Z}_p^2 \setminus \mathsf{Span}(\boldsymbol{b})$. Note that if $\mathsf{id}^\star \notin \mathcal{S}_i$, then the two games $\mathsf{Game}_{3.i-1.2}$ and $\mathsf{Game}_{3.i-1.3}$ are identical. Thus we focus on the case $\mathsf{id}^\star \in \mathcal{S}_i$. In that case we show that $\mathsf{Game}_{3.i-1.3}$ is also identically distributed to $\mathsf{Game}_{3.i-1.2}$ using a statistical argument, which relies on the fact that vectors $\boldsymbol{P}(\mathsf{id}_j)^\top \boldsymbol{b}$ and $\boldsymbol{P}(\mathsf{id}_j)^\top \boldsymbol{d}$ are statistically independent since $\boldsymbol{b}$ and $\boldsymbol{d}$ are linearly independent. The same holds with respect to the matrix $\boldsymbol{P}(0)$. Moreover, since $\mathsf{id}^\star \in \mathcal{S}_i$, the set of values $\{(\boldsymbol{P}(\mathsf{id}_j))_{\mathsf{id}_j \in \mathcal{S}_i}, \boldsymbol{P}(\mathsf{id}^\star)\}$ are statistically independent from the value $\boldsymbol{P}(0)$ — recall that the polynomial P is of degree t; we are using Fact 1 from Section 2.2. Combining these two facts, we know that the vector $\boldsymbol{P}(0)^\top \boldsymbol{d}$ is uniformly random, independent from everything else (challenge ciphertext, public key and other functional secret keys). Thus, it can act as a one-time pad on the value $\boldsymbol{a}^\perp \boldsymbol{x}_\beta^\top \boldsymbol{y}$ that we wish to remove.

$\underline{\mathsf{Game}_{3.i-1.4}}$: is defined as $\mathsf{Game}_{3.i-1.3}$, except the vector $\boldsymbol{d}$ used to compute the i'th queried functional secret key is sampled $\boldsymbol{d} \leftarrow_R \mathbb{Z}_p^2$, instead of uniformly random over $\mathbb{Z}_p^2 \setminus \mathsf{Span}(\boldsymbol{b})$. This is the reverse to the transition from $\mathsf{Game}_{3.i-1.1}$ to $\mathsf{Game}_{3.i-1.2}$. By the same statistical argument, we obtain $\mathsf{Game}_{3.i-1.3} \approx_s \mathsf{Game}_{3.i-1.4}$.

Finally, note that $\mathsf{Game}_{3.i-1.4}$ is the same as $\mathsf{Game}_{3.i}$ except the i'th queried key is computed using $[\![\boldsymbol{d}]\!]_2 \leftarrow_R \mathbb{G}_2^2$ in the former, and $[\![\boldsymbol{b}s_i]\!]_2 \in \mathbb{G}_2^2$ with $s_i \leftarrow_R \mathbb{Z}_p$ in the latter. Therefore, we have $\mathsf{Game}_{3.i-1.4} \approx_c \mathsf{Game}_{3.i}$ by the DDH assumption, which states that $([\![\boldsymbol{b}]\!]_2, [\![\boldsymbol{d}]\!]_2) \approx_c ([\![\boldsymbol{b}]\!]_2, [\![\boldsymbol{b}s_i]\!]_2)$ where $\boldsymbol{b}, \boldsymbol{d} \leftarrow_R \mathbb{Z}_p^2, s_i \leftarrow_R \mathbb{Z}_p$. The former distribution corresponds to $\mathsf{Game}_{3.i-1.4}$, whereas the latter distribution corresponds to $\mathsf{Game}_{3.i}$. Note that this transition is exactly reverse to the transition from $\mathsf{Game}_{3.i-1}$ to $\mathsf{Game}_{3.i-1.1}$. This concludes the proof that $\mathsf{Game}_{3.i-1} \approx_c \mathsf{Game}_{3.i}$ and consequently, that $\mathsf{Game}_3 \approx_c \mathsf{Game}_4$.

Note that in Game_4, the only values that possibly reveal some information about the bit β is the set $\{\boldsymbol{x}_\beta^\top \boldsymbol{y}_i\}$ for all queries $([\![\boldsymbol{y}_i]\!]_2, \mathcal{S}_i)$ such that $\mathsf{id}^\star \notin \mathcal{S}_i$. Since the adversary $\mathcal{A}$ is admissible, we know that for all such values, $\boldsymbol{x}_\beta^\top \boldsymbol{y}_i = \boldsymbol{x}_0^\top \boldsymbol{y}_i = \boldsymbol{x}_1^\top \boldsymbol{y}_i$. In other words, these values do not depend on β and the advantage of $\mathcal{A}$ is 0. □

3.3 Revocations with arbitrary-size identity sets

Our previous construction requires that the size of any identities set $\mathcal{S}$ be exactly t (a pre-established system parameter).

A possible way to relax this limitation is to introduce dummy identities and use them as "fillers", to extend an identity set until it reaches size t. Furthermore, in order to make the secret-key size proportional to the identity set $\mathcal{S}$, we could run different instances of the IPFE for different set-size bounds $t_1, \ldots, t_n$. A secret-key for set $\mathcal{S}$ would then be issued only with respect to the i-th IPFE instance, where t_i is the smallest such that $|\mathcal{S}| \leq t_i$. (Ciphertexts would need to be provided with respect to all IPFE instances). A natural and effective choice for the values of t_i is the set of powers of 2. That way, the ciphertext-size would be increased by a factor of $\log_2$ of the global maximum identity set size. Note that such factor is logairthmic in the security parameter. This technique has already been used in the literature and in particular in the context of ABE, e.g. by Ostrovsky et al. [OSW07, Section 3.3].

4 Generic Construction of MA-ABE from IPFE

We present a modular construction of MA-ABE for non-monotone access structures based on inner-product FE schemes. We show that the resulting MA-ABE is super selectively secure for static corruptions, provided the underlying FE are super selectively secure. The security is proven in the random oracle model.

$\underline{\mathsf{GlobalSetup}(1^\lambda):}$

Generate a pairing group $\mathcal{PG} = (p, \mathbb{G}_1, \mathbb{G}_2, P_1, P_2, \mathbb{G}_{\mathsf{t}}, e) \leftarrow \mathsf{GGen}(1^\lambda)$ and a hash functions $H : \{0,1\}^* \rightarrow \mathbb{G}_2^3$ and return $\mathsf{gp} := (\mathcal{PG}, H)$.

$\underline{\mathsf{AuthSetup}(\mathsf{gp}):}$

Compute $(\mathsf{pk}_\Gamma, \mathsf{msk}_\Gamma) \leftarrow \Gamma.\mathsf{Setup}(1^\lambda, \mathcal{PG})$ and $(\mathsf{pk}_\Sigma, \mathsf{msk}_\Sigma) \leftarrow \Sigma.\mathsf{Setup}(1^\lambda, \mathcal{PG})$. return $\mathsf{pk} = (\mathsf{pk}_\Gamma, \mathsf{pk}_\Sigma)$ and $\mathsf{sk} = (\mathsf{msk}_\Gamma, \mathsf{msk}_\Sigma)$.

$\underline{\mathsf{Enc}\big((\boldsymbol{M} \in \mathbb{Z}_p^{n \times \ell}, \rho : [\ell] \rightarrow \{0,1\}^* \cup (\{\neg\} \cdot \{0,1\}^*)), \{\mathsf{pk}_i\}_{i \in [\nu]}\big):}$

Sample $s \leftarrow_R \mathbb{Z}_p$, and $\{s_j\}_{j \in [\ell]} \leftarrow \mathsf{Share}(\boldsymbol{M}, s)$, $\{u_j\}_{j \in [\ell]} \leftarrow \mathsf{Share}(\boldsymbol{M}, 0)$, $\boldsymbol{a} \leftarrow_R \mathbb{Z}_p^3$. For all $j \in [\ell]$, parse $\mathsf{pk}_{\theta(j)} = (\mathsf{pk}_{\Gamma,\theta(j)}, \mathsf{pk}_{\Sigma,\theta(j)})$, set $\boldsymbol{x}_j = (s_j, u_j \cdot \boldsymbol{a}) \in \mathbb{Z}_p^4$, then

- if $\rho(j) = \mathsf{att}_j$ where $\mathsf{att}_j \in \{0,1\}^*$, then $\mathsf{ct}_j \leftarrow \Gamma.\mathsf{Enc}(\mathsf{pk}_{\Gamma,\theta(j)}, \boldsymbol{x}_j, \mathsf{att}_j)$.
- if $\rho(j) = \neg\mathsf{att}_j$ where $\mathsf{att}_j \in \{0,1\}^*$, then $\mathsf{ct}_j \leftarrow \Sigma.\mathsf{Enc}(\mathsf{pk}_{\Sigma,\theta(j)}, \boldsymbol{x}_j, \mathsf{att}_j)$.

Return $\big(\{\mathsf{ct}_j\}_{j \in [\ell]}, \kappa := \llbracket s \rrbracket_{\mathsf{t}}\big)$.

$\underline{\mathsf{KeyGen}\big(\mathsf{pk}, \mathsf{sk}, \mathsf{gid}, \mathcal{S}\big):}$

Parse $\mathsf{sk} = (\mathsf{msk}_\Gamma, \mathsf{msk}_\Sigma)$. Compute $H(\mathsf{gid}) = \llbracket \boldsymbol{z}_{\mathsf{gid}} \rrbracket_2$, $\mathsf{sk}_\Sigma \leftarrow \Sigma.\mathsf{KeyGen}(\mathsf{msk}_\Sigma, \llbracket 1, \boldsymbol{z}_{\mathsf{gid}} \rrbracket_2, \mathcal{S})$, for all $\mathsf{att}_j \in \mathcal{S}$, $\mathsf{sk}_{\Gamma,j} \leftarrow \Gamma.\mathsf{KeyGen}(\mathsf{msk}_\Gamma, \llbracket 1, \boldsymbol{z}_{\mathsf{gid}} \rrbracket_2, \mathsf{att}_j)$. Return $\mathsf{sk}_{\mathsf{gid},\mathcal{S}} = (\mathsf{sk}_\Sigma, (\mathsf{sk}_{\Gamma,j})_{\mathsf{att}_j \in \mathcal{S}})$.

$\underline{\mathsf{Dec}\,(\mathsf{ct}, \{\mathsf{sk}_{\mathsf{gid},\mathcal{S}_i}\}_i):}$

Parse the ciphertext $\mathsf{ct} = \{\mathsf{ct}_j\}_{j \in [\ell]}$ which contains the description of an access structure $(\boldsymbol{M}, \rho)$. Let $\mathcal{S} = \cup_{i \in [\nu]} \mathcal{S}_i$. Compute $\{\omega_j, \omega'_j\}_{j \in [\ell]}$ such that $\sum_{\rho(j) \in S} \omega_j \boldsymbol{M}_j + \sum_{\rho(j) = \neg\mathsf{att}, \mathsf{att} \notin \mathcal{S}} \omega'_j \boldsymbol{M}_j = \mathbf{1}$. Return $\sum_{\rho(j) \in S} \omega_j \Gamma.\mathsf{Dec}(\mathsf{pk}_{\theta(j)}, \mathsf{ct}_j, \mathsf{sk}_{\mathsf{gid},\mathcal{S}_{\theta(j)}}, \llbracket 1, \boldsymbol{z}_{\mathsf{gid}} \rrbracket_2) + \sum_{\rho(j) = \neg\mathsf{att}, \mathsf{att} \notin \mathcal{S}} \omega'_j \Sigma.\mathsf{Dec}(\mathsf{pk}_{\theta(j)}, \mathsf{ct}_j, \mathsf{sk}_{\mathsf{gid},\mathcal{S}_{\theta(j)}}, \llbracket 1, \boldsymbol{z}_{\mathsf{gid}} \rrbracket_2)$, where $\llbracket \boldsymbol{z}_{\mathsf{gid}} \rrbracket_2 = H(\mathsf{gid})$.

Fig. 3. Construction of Multi-Authority ABE from an ID-IPFE scheme Γ and an IPFE with revocations Σ (for vectors of dimension 4). Recall that θ maps a row $j \in [\ell]$ to the authority that owns the attribute associated to that row.

Correctness. Let $[\![\boldsymbol{z}_{\mathsf{gid}}]\!]_2 := H(\mathsf{gid})$. Observe that, by the correctness of Γ and Σ, we have:

$$\begin{aligned}&\sum_{\rho(j)\in S} \omega_j \Gamma.\mathsf{Dec}(\mathsf{pk}_{\theta(j)}, \mathsf{ct}_j, \mathsf{sk}_{\mathsf{gid},\mathcal{S}_{\theta(j)}}, [\![1, \boldsymbol{z}_{\mathsf{gid}}]\!]_2)\\ &+ \sum_{\rho(j)=\neg\mathsf{att},\mathsf{att}\notin\mathcal{S}} \omega'_j \Sigma.\mathsf{Dec}(\mathsf{pk}_{\theta(j)}, \mathsf{ct}_j, \mathsf{sk}_{\mathsf{gid},\mathcal{S}_{\theta(j)}}, [\![1, \boldsymbol{z}_{\mathsf{gid}}]\!]_2)\\ &= \sum_{\rho(j)\in S} \omega_j \left[\!\left[s_j + \boldsymbol{a}^\top \boldsymbol{z}_{\mathsf{gid}} u_j\right]\!\right]_{\mathsf{t}} + \sum_{\rho(j)=\neg\mathsf{att},\mathsf{att}\notin\mathcal{S}} \omega'_j \left[\!\left[s_j + \boldsymbol{a}^\top \boldsymbol{z}_{\mathsf{gid}} u_j\right]\!\right]_{\mathsf{t}}\\ &= \left[\!\left[s + \boldsymbol{a}^\top \boldsymbol{z}_{\mathsf{gid}} \cdot 0\right]\!\right]_{\mathsf{t}} = \kappa \enspace .\end{aligned}$$

Theorem 2 (Super-selective security). *The scheme from Fig. 3, is a super-selectively secure MA-ABE with static corruption in the random oracle model, assuming the schemes Γ and Σ are super-selectively secure and the DDH assumption holds in $\mathbb{G}_2$.*

Combining with the existence of an ID-IPFE selectively secure under SXDH (from [ACGU20]) and Theorem 1 (the existence of selectively secure IPFE with revocations from SXDH) and noting that selective security implies super-selective security, we obtain the following corollary.

Corollary 1. *There exists a super-selectively secure MA-ABE with static corruptions from SXDH.*

We now proceed to prove the theorem.

Proof. We prove security via a sequence of hybrid games. We highlight in red the changes from one hybrid to the next when relevant.

$\underline{\mathsf{Game}_0:}$ The first game corresponds to the super-selective security game for MA-ABE with static corruptions, defined in Section 2.6. We recall it here for completeness. We call $\mathcal{A}$ the admissible adversary. First, $\mathcal{A}$ receives the global parameters $\mathsf{gp} = (\Gamma.\mathsf{gp}, H)$. Then, it can query its oracle $\mathcal{O}_{\mathsf{create}}$ that creates a new (honest) authority with an associated $(\mathsf{pk}, \mathsf{sk})$ pair when invoked, adds pk to the set of honest authorities denoted by $\mathcal{S}_{\mathsf{hon}}$ and returns pk to $\mathcal{A}$. Then, $\mathcal{A}$ sends $(\boldsymbol{M}, \rho, \Pi_{\mathsf{hon}}, \Pi_{\mathsf{corr}})$ to its challenger, where $\boldsymbol{M} \in \mathbb{Z}_p^{n\times\ell}$, $\rho : [\ell] \to \mathbb{Z}_p$ is an access structure with attributes owned by the authorities in the set $\Pi = \Pi_{\mathsf{hon}} \cup \Pi_{\mathsf{corr}}$. Here, Π_{hon} is a set of honest authorities' public keys, that is, $\Pi_{\mathsf{hon}} \subseteq \mathcal{S}_{\mathsf{hon}}$, and Π_{corr} is a set of authorities' public key created by $\mathcal{A}$ itself (and not via $\mathcal{O}_{\mathsf{create}}$). Because $\mathcal{A}$ is free to create these public keys however it wants (potentially maliciously), these are referred to as corrupted authorities. Note that $\mathcal{A}$ cannot query its oracle $\mathcal{O}_{\mathsf{corr}}$, since we assume only static corruptions here. We write $\Pi = \{\mathsf{pk}_1, \ldots, \mathsf{pk}_\nu\}$, and we define $\theta : [\ell] \to [\nu]$, which maps each column $j \in [\ell]$ to the authority that owns the attribute associated with that column.

Afterwards, the adversary can query its oracle $\mathcal{O}_{\mathsf{KeyGen}}$ on inputs $\mathsf{pk} \in \mathcal{S}_{\mathsf{hon}}$ associated with the secret key $\mathsf{sk} = (\mathsf{msk}_\Sigma, \mathsf{msk}_\Gamma)$ and $\mathcal{S} \subset \mathcal{U}_{\mathsf{pk}}$, which computes $\mathsf{sk}_\Sigma \leftarrow \Sigma.\mathsf{KeyGen}(\mathsf{msk}_\Sigma, [\![1, \boldsymbol{z}_{\mathsf{gid}}]\!]_2, \mathcal{S})$ and for all attributes $\mathsf{att}_j \in \mathcal{S}$, it computes $\mathsf{sk}_{\Gamma,j} \leftarrow \Gamma.\mathsf{KeyGen}(\mathsf{msk}_\Gamma, [\![1, \boldsymbol{z}_{\mathsf{gid}}]\!]_2, \mathsf{att}_j)$, where $[\![\boldsymbol{z}_{\mathsf{gid}}]\!]_2 = H(\mathsf{gid})$. It returns $\mathsf{sk}_{\mathsf{gid},\mathcal{S}} = (\mathsf{sk}_\Sigma, (\mathsf{sk}_{\Gamma,j})_{\mathsf{att}_j \in \mathcal{S}})$ to the adversary $\mathcal{A}$.

At this point, the challenger samples $s \leftarrow_R \mathbb{Z}_p$ and computes $(s_1, \ldots, s_\ell) \leftarrow \mathsf{Share}(\boldsymbol{M}, s)$, $(u_1, \ldots, u_\ell) \leftarrow \mathsf{Share}(\boldsymbol{M}, 0)$[4], $\boldsymbol{a} \leftarrow_R \mathbb{Z}_p^3$, $\kappa_0 = [\![s]\!]_{\mathsf{t}}$, $\kappa_1 \leftarrow_R \mathbb{G}_{\mathsf{t}}$, $\beta \leftarrow_R \{0,1\}$, for all $j \in [\ell]$, $\boldsymbol{x}_j = (s_j, u_j \cdot \boldsymbol{a}) \in \mathbb{Z}_p^4$, and

- if $\rho(j) = \mathsf{att}_j$ where $\mathsf{att}_j \in \{0,1\}^*$, then $\mathsf{ct}_j \leftarrow \Gamma.\mathsf{Enc}(\mathsf{pk}_{\Gamma,\theta(j)}, \boldsymbol{x}_i, \rho(j))$,
- if $\rho(j) = \neg\mathsf{att}_j$ where $\mathsf{att}_j \in \{0,1\}^*$, then $\mathsf{ct}_j \leftarrow \Sigma.\mathsf{Enc}(\mathsf{pk}_{\Sigma,\theta(j)}, \boldsymbol{x}_i, \rho(j))$.

It sets $\mathsf{ct}^\star = \{\mathsf{ct}_j\}_{j\in[\ell]}$ and returns $(\mathsf{ct}^\star, \kappa_\beta)$ to $\mathcal{A}$. Finally, $\mathcal{A}$ outputs a guess $\beta' \in \{0,1\}$. Recall that $\mathcal{A}$ is admissible, which means it cannot compute κ_0 from $\mathsf{ct}^\star$ simply by correctness of the scheme with the user secret keys it queried and the secret key of the corrupted authorities (see Section 2.6 for more details). The experiment outputs 1 if $\beta = \beta'$, 0 otherwise.

In the following hybrids, we use the following dual basis: first, we choose a random basis $(\boldsymbol{a}_1|\boldsymbol{a}_2|\boldsymbol{a}_3) \in \mathbb{Z}_p^{3\times 3}$ of $\mathbb{Z}_p^3$ such that $\boldsymbol{a} = r_1\boldsymbol{a}_1$ for $r_1 \leftarrow_R \mathbb{Z}_p^*$ (recall that the vector $\boldsymbol{a}$ is sampled to produce the challenge ciphertext). Strictly speaking, such a basis exists only when $\boldsymbol{a} \neq \boldsymbol{0}$. Since $\boldsymbol{a}$ is sampled uniformly at random over $\mathbb{Z}_p^3$, it is different from $\boldsymbol{0}$ with overwhelming probability. Thus, we implicitly assume $\boldsymbol{a}$ is sampled uniformly over $\mathbb{Z}_p^3 \setminus \{\boldsymbol{0}\}$ in the proof (this only changes the distribution by a negligible statistical distance). Then, we denote by $(\boldsymbol{a}_1^*|\boldsymbol{a}_2^*|\boldsymbol{a}_3^*) \in \mathbb{Z}_p^{3\times 3}$ its dual basis, that is, such that for all $i, j \in \{1,2,3\}$, $\boldsymbol{a}_i^\top \boldsymbol{a}_j^* = 0$ if $i \neq j$ and $\boldsymbol{a}_i^\top \boldsymbol{a}_j^* = 1$ if $i = j$. We make use of the following assumptions relative to the pairing groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ and the random dual basis $(\boldsymbol{a}_1|\boldsymbol{a}_2|\boldsymbol{a}_3)$ and $(\boldsymbol{a}_1^*|\boldsymbol{a}_2^*|\boldsymbol{a}_3^*)$.

Assumption 1. $\{\boldsymbol{v} \leftarrow_R \mathbb{Z}_p^3 : ([\![\boldsymbol{a}_1]\!]_1, [\![\boldsymbol{v}]\!]_2)\} \approx_c \{\boldsymbol{v} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*) : ([\![\boldsymbol{a}_1]\!]_1, [\![\boldsymbol{v}]\!]_2)\}$.

This assumption is known to be implied by the DDH assumption in $\mathbb{G}_2$ (see for instance [Lew12]).

$\underline{\mathsf{Game}_1}$: is the same as Game_0 except that the outputs of the hash function are computed as follows: for all gid, $H(\mathsf{gid}) = [\![\boldsymbol{z}_{\mathsf{gid}}]\!]_2$ where $\boldsymbol{z}_{\mathsf{gid}} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*)$. We have $\mathsf{Game}_0 \approx_c \mathsf{Game}_1$ by **Assumption 1**. Technically, we need to use this assumption for each query of $\mathcal{A}$ to the hash function H (modeled as a random oracle) using a hybrid argument.

$\underline{\mathsf{Game}_2}$: is the same as Game_1, except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + v_j \boldsymbol{a}_3), \rho(j))$, for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, where $v_j = (\gamma, \boldsymbol{v})^\top \boldsymbol{M}_j$, $\boldsymbol{v} \leftarrow_R \mathbb{Z}_p^{n-1}$, and $\gamma \leftarrow_R \mathbb{Z}_p$. That is, the v_j are shares of a random value γ. Recall that $\boldsymbol{a}_1, \boldsymbol{a}_3 \in \mathbb{Z}_p^3$ are vectors part of the basis $(\boldsymbol{a}_1|\boldsymbol{a}_2|\boldsymbol{a}_3)$

[4] See Fig. 1 for the definition of the algorithm Share.

and $\boldsymbol{a} = r_1\boldsymbol{a}_1$ where $r_1 \leftarrow_R \mathbb{Z}_p^*$. The shares s_j and u_j are computed as before. For all $j \in [\ell]$ such that $\theta(j) \in \mathit{\Pi}_{\mathsf{corr}}$, the vector $\boldsymbol{x}_j$ are as before. The challenge ciphertext is set to be $(\{\mathsf{ct}_j\}_{j\in[\ell]}, \kappa_\beta)$, where κ_β is computed as before. We argue that $\mathsf{Game}_1 \approx_c \mathsf{Game}_2$ using the super-selective security of $\mathit{\Gamma}$ and $\mathit{\Sigma}$, since the extra red vector $(0, v_j\boldsymbol{a}_3)$ is orthogonal to the vectors $[\![1, \boldsymbol{z}_{\mathsf{gid}}]\!]_2$ from the user secret keys. This is because for all queried gid, $\boldsymbol{z}_{\mathsf{gid}} \in \mathsf{Span}(\boldsymbol{a}_1^*)$ and $\boldsymbol{a}_3^\top \boldsymbol{a}_1^* = 0$.

$\underline{\mathsf{Game}_3}$: is the same as Game_2, except that the outputs of the hash function are computed as follows: for all gid, $H(\mathsf{gid}) = [\![\boldsymbol{a}_1^* r_{\mathsf{gid}} + \boldsymbol{a}_3^*]\!]_2$, where $r_{\mathsf{gid}} \leftarrow_R \mathbb{Z}_p$. We prove that $\mathsf{Game}_2 \approx_c \mathsf{Game}_3$ in Lemma 2.

$\underline{\mathsf{Game}_4}$: is the same as Game_3, except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s'_j, u_j r_1 \boldsymbol{a}_1 + v'_j \boldsymbol{a}_3)$ for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, where $s'_j = (s + \gamma, \boldsymbol{w})^\top \boldsymbol{M}_j$ and $v'_j = (0, \boldsymbol{v})^\top \boldsymbol{M}_j$. That is, the s'_j are now shares of $s + \gamma$ instead of s and the v'_j are now shares of 0 instead of γ. The shares u_j are computed as before. We argue that $\mathsf{Game}_3 \approx_c \mathsf{Game}_4$ thanks to the super-selective security of $\mathit{\Gamma}$ and $\mathit{\Sigma}$. Indeed, for all $j \in [\ell]$ and all queried gid, we have $(s'_j, u_j r_1 \boldsymbol{a}_1 + v'_j \boldsymbol{a}_3)^\top (1, \boldsymbol{a}_1^* r_{\mathsf{gid}} + \boldsymbol{a}_3^*) = (s+\gamma, \boldsymbol{w})^\top \boldsymbol{M}_j + r_1 r_{\mathsf{gid}} (0, \boldsymbol{u})^\top \boldsymbol{M}_j + (0, \boldsymbol{v})^\top \boldsymbol{M}_j = (s, \boldsymbol{w})^\top \boldsymbol{M}_j + r_1 r_{\mathsf{gid}} (0, \boldsymbol{u})^\top \boldsymbol{M}_j + (\gamma, \boldsymbol{v})^\top \boldsymbol{M}_j = s_j + r_1 r_{\mathsf{gid}} u_j + v_j = (s_j, u_j r_1 \boldsymbol{a}_1 + v_j \boldsymbol{a}_3)^\top (1, \boldsymbol{a}_1^* r_{\mathsf{gid}} + \boldsymbol{a}_3^*)$, just as in Game_3. That is, the change of the vectors encrypted under $\mathit{\Gamma}$ from Game_3 to Game_4 preserves the value of the inner product.

Finally, to conclude the proof, we show that in Game_4, the advantage of $\mathcal{A}$ is 0. This comes from the fact that the value $\kappa_0 = [\![s]\!]_{\mathsf{t}}$ is uniformly random, independent of the rest of the adversary's view. Indeed, the only place where the value s appears is in the challenge ciphertext, in the vectors $\boldsymbol{x}_j$ encrypted under $\mathit{\Gamma}$ or $\mathit{\Sigma}$. For all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, the vector $\boldsymbol{x}_j$ is of the form $\boldsymbol{x}_j = (s'_j, u_j r_1 \boldsymbol{a}_1 + v'_j \boldsymbol{a}_3))$ where s'_j is of the form $s'_j = (s+\gamma, \boldsymbol{w})^\top \boldsymbol{M}_j$ for all $j \in [\ell]$. That is, the values s'_j are shares of the secret $s + \gamma$. But the value $\gamma \leftarrow_R \mathbb{Z}_p$ is independent of the rest of the adversary's view, thus it acts as a one-time pad on s. Consequently, $\boldsymbol{x}_j$ is independent of the value s. For all $j \in [\ell]$ such that $\theta(j) \in \mathit{\Pi}_{\mathsf{corr}}$, we have $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + v_j \boldsymbol{a}_3))$, where the values s_j are shares of the secret s. But because the adversary $\mathcal{A}$ is admissible, we know that the shares $\{s_j\}_{j\in[\ell], \theta(j)\in \mathit{\Pi}_{\mathsf{corr}}}$ are independent of s, by security of the MSP. Thus, both κ_0 and κ_1 are uniformly random independent of everything else, the view of the adversary does not depend on the bit β; its advantage is 0. □

Now we state and prove the lemma used in the proof above. Its proof relies on the assumptions below, which are known to be implied by DDH in $\mathbb{G}_2$ (see for instance [Lew12]).

Lemma 2. *We have* $\mathsf{Game}_2 \approx_c \mathsf{Game}_3$ *assuming the super-selective security of* $\mathit{\Gamma}$ *and* $\mathit{\Sigma}$*, and the SXDH assumption.*

To prove the lemma, we rely on the following assumptions, which are known to be implied by DDH in $\mathbb{G}_2$ (see for instance [Lew12]).

Assumption 2.

$$\{\boldsymbol{v} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*), r_1, r_2 \leftarrow_R \mathbb{Z}_p^* : ([\![r_1\boldsymbol{a}_1 + r_2\boldsymbol{a}_2]\!]_1, [\![\boldsymbol{a}_1]\!]_1, [\![\boldsymbol{a}_3]\!]_1, [\![\boldsymbol{a}_1^*]\!]_2, [\![\boldsymbol{a}_3^*]\!]_2, [\![\boldsymbol{v}]\!]_2)\}$$
$$\approx_c \{\boldsymbol{v} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_2^*), r_1, r_2 \leftarrow_R \mathbb{Z}_p^* : ([\![r_1\boldsymbol{a}_1 + r_2\boldsymbol{a}_2]\!]_1, [\![\boldsymbol{a}_1]\!]_1, [\![\boldsymbol{a}_3]\!]_1, [\![\boldsymbol{a}_1^*]\!]_2, [\![\boldsymbol{a}_3^*]\!]_2, [\![\boldsymbol{v}]\!]_2)\}\,.$$

Assumption 3.

$$\{\boldsymbol{v} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_2^*), r \leftarrow_R \mathbb{Z}_p : ([\![\boldsymbol{a}_1]\!]_1, [\![r\boldsymbol{a}_2 + \boldsymbol{a}_3]\!]_1, [\![\boldsymbol{a}_1^*]\!]_2, [\![\boldsymbol{a}_3^*]\!]_2, [\![\boldsymbol{v}]\!]_2)\}$$
$$\approx_c \{\boldsymbol{v} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_2^*, \boldsymbol{a}_3^*), r \leftarrow_R \mathbb{Z}_p : ([\![\boldsymbol{a}_1]\!]_1, [\![r\boldsymbol{a}_2 + \boldsymbol{a}_3]\!]_1, [\![\boldsymbol{a}_1^*]\!]_2, [\![\boldsymbol{a}_3^*]\!]_2, [\![\boldsymbol{v}]\!]_2)\}\,.$$

Proof. To prove the lemma, we introduce the following hybrid games for all $i \in \{0, \dots, q\}$ where $q \in \mathbb{N}$ denotes the number of distinct gid queried via $\mathcal{O}_{\mathsf{KeyGen}}$: $\mathsf{Game}_{2.i}$ is like Game_2, except that for the first i'th gid, $\mathcal{O}_{\mathsf{KeyGen}}$ behaves like in Game_3. Namely, for the first i'th gid queried to $\mathcal{O}_{\mathsf{KeyGen}}$, the oracle uses $H(\mathsf{gid}) = [\![\boldsymbol{a}_1^* r_{\mathsf{gid}} + \boldsymbol{a}_3^*]\!]_2$, whereas it uses $H(\mathsf{gid}) = [\![\boldsymbol{a}_1^* r_{\mathsf{gid}}]\!]_2$ for the last $q - i$ queries. It is clear by definition of the games that $\mathsf{Game}_{2.0} = \mathsf{Game}_2$ and $\mathsf{Game}_{2.q} = \mathsf{Game}_3$. We prove that for all $i \in [q]$, $\mathsf{Game}_{2.i-1} \approx_c \mathsf{Game}_{2.i}$. To do so, we use the following hybrid games.

$\mathsf{Game}_{2.i-1.1}$: is the same as $\mathsf{Game}_{2.i-1}$, except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + u_j r_2 \boldsymbol{a}_2 + v_j \boldsymbol{a}_3)$ for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, where $r_2 \leftarrow_R \mathbb{Z}_p^*$. We argue that $\mathsf{Game}_{2.i-1} \approx_c \mathsf{Game}_{2.i-1.1}$ thanks to the super-selective security of Γ and Σ. Indeed, for all $j \in [\ell]$ and all queried gid, we have $\boldsymbol{z}_{\mathsf{gid}} \in \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_3^*)$, thus $(s_j, u_j r_1 \boldsymbol{a}_1 + u_j r_2 \boldsymbol{a}_2 + v_j \cdot \boldsymbol{a}_3)^\top (1, \boldsymbol{z}_{\mathsf{gid}}) = (s_j, u_j r_1 \boldsymbol{a}_1 + v_j \cdot \boldsymbol{a}_3)^\top (1, \boldsymbol{z}_{\mathsf{gid}})$, just as in game $\mathsf{Game}_{2.i-1}$, since $\boldsymbol{a}_2^\top \boldsymbol{a}_1^* = \boldsymbol{a}_2^\top \boldsymbol{a}_3^* = 0$.

$\mathsf{Game}_{2.i-1.2}$: is the same as $\mathsf{Game}_{2.i-1.1}$ except that the output of the hash function on the i'th queried global identifier, which we denote by gid_i, is computed as follows: $H(\mathsf{gid}_i) = [\![\boldsymbol{z}_{\mathsf{gid}_i}]\!]_2$ where $\boldsymbol{z}_{\mathsf{gid}_i} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_2^*)$, as opposed to uniformly random over $\mathsf{Span}(\boldsymbol{a}_1^*)$ in $\mathsf{Game}_{2.i-1.1}$. We have $\mathsf{Game}_{2.i-1.1} \approx_c \mathsf{Game}_{2.i-1.2}$ by **Assumption 2**.

$\mathsf{Game}_{2.i-1.3}$: is the same as $\mathsf{Game}_{2.i-1.2}$ except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + r_j \boldsymbol{a}_2 + v_j \boldsymbol{a}_3)$ for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, where $r_j = (0, \boldsymbol{r})^\top \boldsymbol{M}_j$, and $\boldsymbol{r} \leftarrow_R \mathbb{Z}_p^{n-1}$. We have that $\mathsf{Game}_{2.i-1.2} \approx_c \mathsf{Game}_{2.i-1.3}$ from the DDH assumption in $\mathbb{G}_1$, which implies that $\{r_2 \leftarrow_R \mathbb{Z}_p^*, \boldsymbol{u} \leftarrow_R \mathbb{Z}_p^{n-1} : ([\![\boldsymbol{u}]\!]_1, [\![r_2 \boldsymbol{u}]\!]_1)\} \approx_c \{\boldsymbol{u}, \boldsymbol{r} \leftarrow_R \mathbb{Z}_p^{n-1} : ([\![\boldsymbol{u}]\!]_1, [\![\boldsymbol{r}]\!]_1)\}$ [5]

$\mathsf{Game}_{2.i-1.4}$: is the same as $\mathsf{Game}_{2.i-1.3}$ except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + r_j \boldsymbol{a}_2 + \eta_j \boldsymbol{a}_2 + v_j \boldsymbol{a}_3)$ for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, where the value η_j is defined as $\eta(1, \boldsymbol{w}_{\mathsf{gid}_i})^\top \boldsymbol{M}_j$, where $\eta \leftarrow_R \mathbb{Z}_p$ and $\boldsymbol{w}_{\mathsf{gid}_i}$ is a vector such that $(1, \boldsymbol{w}_{\mathsf{gid}_i})^\top \boldsymbol{M}_j = 0$ for all $j \in [\ell]$ such that

[5] Strictly speaking, the DDH as per Definition 3 is stated with $r_2 \leftarrow_R \mathbb{Z}_p$, not $r_2 \leftarrow_R \mathbb{Z}_p^*$ used here. This makes no difference, however, since the two distributions are within negligible statistical distance.

$\rho(j) \in \mathcal{S}_{\mathsf{gid}_i}$ or $(\rho(j) = \neg\mathsf{att}_j$ and $\mathsf{att}_j \in \{0,1\}^* \setminus \mathcal{S}_{\mathsf{gid}_i})$. The set $\mathcal{S}_{\mathsf{gid}_i}$ is defined as $\mathcal{S}_{\mathsf{gid}_i} = \cup_{\mathsf{pk} \in \mathcal{S}_{\mathsf{hon}}, (\mathsf{pk}, \mathsf{gid}_i, \mathcal{S}) \in \mathcal{Q}_{\mathsf{KeyGen}}} \mathcal{S}$. We know that $\mathcal{S}_{\mathsf{gid}_i}$ does not satisfy the access structure $(\boldsymbol{M}, \rho)$ of the challenge ciphertext, because the adversary is admissible. Thus, by security of the access structure (Lemma 1), we know that such a vector $\boldsymbol{w}_{\mathsf{gid}_i} \in \mathbb{Z}_p^{n-1}$ exists. Note that we crucially rely on the selectivity here, since the vector $\boldsymbol{w}_{\mathsf{gid}_i}$ used in the challenge ciphertext depends on attributes queried to $\mathcal{O}_{\mathsf{KeyGen}}$. The fact that $\mathsf{Game}_{2.i-1.4} \approx_c \mathsf{Game}_{2.i-1.3}$ follows from the super-selective security of $\mathit{\Gamma}$ and $\mathit{\Sigma}$. Indeed, the extra red component $\eta_j \boldsymbol{a}_2$ encrypted under $\mathit{\Gamma}$ or $\mathit{\Sigma}$ never interacts with the vectors used to produce user secret keys. Namely, for all $\mathsf{gid} \neq \mathsf{gid}_i$, we have $H(\mathsf{gid}) = \llbracket \boldsymbol{z}_{\mathsf{gid}} \rrbracket_2$ with $\boldsymbol{z}_{\mathsf{gid}} \in \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_3^*)$ so $(0, \eta_j \boldsymbol{a}_2)^\top (1, \boldsymbol{z}_{\mathsf{gid}}) = 0$. For $\mathsf{gid} = \mathsf{gid}_i$, we argue that for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, either $\eta_j = 0$, or the extra $\eta_j \boldsymbol{a}_2$ can be added thanks to the super-selective security of $\mathit{\Gamma}$ and $\mathit{\Sigma}$. When $\rho(j) \in \mathcal{S}_{\mathsf{gid}_i}$ or $\rho(j) = \neg\mathsf{att}$ with $\mathsf{att} \in \{0,1\}^* \setminus \mathcal{S}_{\mathsf{gid}_i}$, we know that $\eta_j = 0$. When $\rho(j)$ is not of this form, then we know that none of the functional secret keys generated by $\mathcal{O}_{\mathsf{KeyGen}}$ on gid_i decrypt the ciphertext ct_j. Thus, we can conclude using the super-selective security of $\mathit{\Sigma}$ and $\mathit{\Gamma}$.

$\mathsf{Game}_{2.i-1.5}$: is the same as $\mathsf{Game}_{2.i-1.4}$ except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + \eta'_j \boldsymbol{a}_2 + v_j \boldsymbol{a}_3)$ where $\eta'_j = (\eta, \boldsymbol{r})^\top \boldsymbol{M}_j$, for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$. The fact that $\mathsf{Game}_{2.i-1.5} = \mathsf{Game}_{2.i-1.4}$ follows from the fact a uniformly random vector $\boldsymbol{r} \leftarrow_R \mathbb{Z}_p^{n-1}$ is distributed identically to an offset $\boldsymbol{x} \in \mathbb{Z}_p^{n-1}$ plus a uniformly random vector $\boldsymbol{r} \leftarrow_R \mathbb{Z}_p^{n-1}$. This is true no matter the value of $\boldsymbol{x}$, as long as $\boldsymbol{r}$ is sampled independently of $\boldsymbol{x}$. So, the following distributions are equals: $\{r_j + \eta_j\}_{j \in [\ell]} = \{(0, \boldsymbol{r})^\top \boldsymbol{M}_j + \eta(1, \boldsymbol{w}_{\mathsf{gid}_i})^\top \boldsymbol{M}_j\}_{j \in [\ell]} = \{(\eta, \boldsymbol{r} + \eta \boldsymbol{w}_{\mathsf{gid}_i})^\top \boldsymbol{M}_j\}_{j \in [\ell]} \equiv \{(\eta, \boldsymbol{r})^\top \boldsymbol{M}_j\}_{j \in [\ell]} = \{\eta'_j\}_{j \in [\ell]}$. This first distribution corresponds to $\mathsf{Game}_{2.i-1.4}$, whereas the last distribution corresponds to $\mathsf{Game}_{2.i-1.5}$.

$\mathsf{Game}_{2.i-1.6}$: is the same as $\mathsf{Game}_{2.i-1.5}$ except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + r'_j \boldsymbol{a}_2 + v_j \boldsymbol{a}_3)$ or all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, where $r'_j = r(\gamma, \boldsymbol{v})^\top \boldsymbol{M}_j$, $r \leftarrow_R \mathbb{Z}_p$. Recall that $\gamma \in \mathbb{Z}_p$ and $\boldsymbol{v} \in \mathbb{Z}_p^{n-1}$ are used to compute the shares v_j, namely $v_j = (\gamma, \boldsymbol{v})^\top \boldsymbol{M}_j$. We argue that $\mathsf{Game}_{2.i-1.5} \approx_c \mathsf{Game}_{2.i-1.6}$ using the DDH assumption in $\mathbb{G}_1$, which implies that $\{\boldsymbol{r}, \boldsymbol{v} \leftarrow_R \mathbb{Z}_p^{n-1}, \eta, \gamma \leftarrow_R \mathbb{Z}_p : (\llbracket \eta \rrbracket_1, \llbracket \boldsymbol{r} \rrbracket_1, \llbracket \gamma \rrbracket_1, \llbracket \boldsymbol{v} \rrbracket_1)\} \approx_c \{\boldsymbol{v} \leftarrow_R \mathbb{Z}_p^{n-1}, r, \gamma \leftarrow_R \mathbb{Z}_p : (\llbracket r\gamma \rrbracket_1, \llbracket r\boldsymbol{v} \rrbracket_1, \llbracket \gamma \rrbracket_1, \llbracket \boldsymbol{v} \rrbracket_1)\}$.

$\mathsf{Game}_{2.i-1.7}$: is the same as $\mathsf{Game}_{2.i-1.6}$ except that the output of the hash function on the i'th queried global identifier, which we denote by gid_i, is computed as follows: $H(\mathsf{gid}_i) = \llbracket \boldsymbol{z}_{\mathsf{gid}_i} + \boldsymbol{a}_3^* \rrbracket_2$ where $\boldsymbol{z}_{\mathsf{gid}_i} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_2^*)$. We have $\mathsf{Game}_{2.i-1.6} \approx_c \mathsf{Game}_{2.i-1.7}$ by **Assumption 3**. Indeed, we have $\{\boldsymbol{z}_{\mathsf{gid}_i} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_2^*) : \llbracket \boldsymbol{z}_{\mathsf{gid}_i} \rrbracket_2\} \approx_c \{\boldsymbol{z}_{\mathsf{gid}_i} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_2^*, \boldsymbol{a}_3^*) : \llbracket \boldsymbol{z}_{\mathsf{gid}_i} \rrbracket_2\} \equiv \{\boldsymbol{z}_{\mathsf{gid}_i} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_2^*, \boldsymbol{a}_3^*) : \llbracket \boldsymbol{z}_{\mathsf{gid}_i} + \boldsymbol{a}_3^* \rrbracket_2\} \approx_c \{\boldsymbol{z}_{\mathsf{gid}_i} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_2^*) : \llbracket \boldsymbol{z}_{\mathsf{gid}_i} + \boldsymbol{a}_3^* \rrbracket_2\}$, where the $\approx_c$ follows from **Assumption 3**. The first distribution corresponds to $\mathsf{Game}_{2.i-1.6}$, whereas the last distribution corresponds to $\mathsf{Game}_{2.i-1.7}$. Note that for readability we omit the other values $(\llbracket \boldsymbol{a}_1 \rrbracket_1, \llbracket r\boldsymbol{a}_2 + \boldsymbol{a}_3 \rrbracket_1, \llbracket \boldsymbol{a}_1^* \rrbracket_2, \llbracket \boldsymbol{a}_3^* \rrbracket_2)$ present in the output of all distributions. These values are sufficient to generate the entire adversary's view.

$\underline{\mathsf{Game}_{2.i-1.8}}$: is the same as $\mathsf{Game}_{2.i-1.7}$ except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + \eta'_j \boldsymbol{a}_2 + v_j \boldsymbol{a}_3)$ for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, where $\eta'_j = (\eta, \boldsymbol{r})^\top \boldsymbol{M}_j$, $\eta \leftarrow_R \mathbb{Z}_p$, $\boldsymbol{r} \leftarrow_R \mathbb{Z}_p^{n-1}$. This is the reverse of the transition from $\mathsf{Game}_{2.i-1.5}$ and $\mathsf{Game}_{2.i-1.6}$. We have $\mathsf{Game}_{2.i-1.7} \approx_c \mathsf{Game}_{2.i-1.8}$ using the DDH assumption in $\mathbb{G}_1$, which implies that $\{r, \gamma \leftarrow_R \mathbb{Z}_p, \boldsymbol{v} \leftarrow_R \mathbb{Z}_p^{n-1} : (\llbracket r\gamma \rrbracket_1, \llbracket r\boldsymbol{v} \rrbracket_1, \llbracket \gamma \rrbracket_1, \llbracket \boldsymbol{v} \rrbracket_1)\} \approx_c \{\eta, \gamma \leftarrow_R \mathbb{Z}_p, \boldsymbol{r}, \boldsymbol{v} \leftarrow_R \mathbb{Z}_p^{n-1} : (\llbracket \eta \rrbracket_1, \llbracket \boldsymbol{r} \rrbracket_1, \llbracket \gamma \rrbracket_1, \llbracket \boldsymbol{v} \rrbracket_1)\}$.

$\underline{\mathsf{Game}_{2.i-1.9}}$: is the same as $\mathsf{Game}_{2.i-1.8}$ except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + r_j \boldsymbol{a}_2 + \eta_j + v_j \boldsymbol{a}_3)$ for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, where $r_j = (0, \boldsymbol{r})^\top \boldsymbol{M}_j$, $\eta_j = \eta(1, \boldsymbol{w}_{\mathsf{gid}_i})^\top \boldsymbol{M}_j$, $\eta \leftarrow_R \mathbb{Z}_p$ and $\boldsymbol{w}_{\mathsf{gid}_i}$ is defined as before. This is the reverse of the transition from $\mathsf{Game}_{2.i-1.4}$ and $\mathsf{Game}_{2.i-1.5}$. The fact that $\mathsf{Game}_{2.i-1.8} = \mathsf{Game}_{2.i-1.9}$ follows from the fact a uniformly random vector $\boldsymbol{r} \leftarrow_R \mathbb{Z}_p^{n-1}$ is distributed identically to an offset $\boldsymbol{x} \in \mathbb{Z}_p^{n-1}$ plus a uniformly random vector $\boldsymbol{r} \leftarrow_R \mathbb{Z}_p^{n-1}$, as long as $\boldsymbol{r}$ is sampled independently of $\boldsymbol{x}$. So, the following distributions are equals: $\{\eta'_j\}_{j \in [\ell]} = \{(\eta, \boldsymbol{r})^\top \boldsymbol{M}_j\}_{j \in [\ell]} \equiv \{(\eta, \boldsymbol{r} + \eta \boldsymbol{w}_{\mathsf{gid}_i})^\top \boldsymbol{M}_j\}_{j \in [\ell]} = \{(0, \boldsymbol{r})^\top \boldsymbol{M}_j + \eta(1, \boldsymbol{w}_{\mathsf{gid}_i})^\top \boldsymbol{M}_j\}_{j \in [\ell]} = \{r_j + \eta_j\}_{j \in [\ell]}$. This first distribution corresponds to $\mathsf{Game}_{2.i-1.8}$, whereas the last distribution corresponds to $\mathsf{Game}_{2.i-1.9}$.

$\underline{\mathsf{Game}_{2.i-1.10}}$: is the same as $\mathsf{Game}_{2.i-1.9}$ except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + r_j \boldsymbol{a}_2 + v_j \boldsymbol{a}_3)$ for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, where $r_j = (0, \boldsymbol{r})^\top \boldsymbol{M}_j$, $\boldsymbol{r} \leftarrow_R \mathbb{Z}_p^{n-1}$. This is the reverse of the transition from $\mathsf{Game}_{2.i-1.3}$ and $\mathsf{Game}_{2.i-1.4}$. The fact that $\mathsf{Game}_{2.i-1.9} \approx_c \mathsf{Game}_{2.i-1.10}$ follows from the super-selective security of $\varGamma$ and $\varSigma$. Indeed, the component $\eta_j \boldsymbol{a}_2$ encrypted under $\varGamma$ and $\varSigma$ in $\mathsf{Game}_{2.i-1.9}$ never interacts with the vectors used to produce user secret keys. Namely, for all $\mathsf{gid} \neq \mathsf{gid}_i$, we have $H(\mathsf{gid}) = \llbracket \boldsymbol{z}_{\mathsf{gid}} \rrbracket_2$ with $\boldsymbol{z}_{\mathsf{gid}} \in \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_3^*)$ so $(0, \eta_j \boldsymbol{a}_2)^\top (1, \boldsymbol{z}_{\mathsf{gid}}) = 0$. For $\mathsf{gid} = \mathsf{gid}_i$, we know that all queries $(\mathsf{pk}, \mathsf{gid}_i, \mathcal{S})$ to $\mathcal{O}_{\mathsf{KeyGen}}$ are such that $\mathcal{S} \in \mathcal{S}_{\mathsf{gid}_i}$ (by definition of the set $\mathcal{S}_{\mathsf{gid}_i}$), and, as argued before, we know that for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, either $\eta_j = 0$ or ct_j cannot be decrypted by the functional secret keys generated by $\mathcal{O}_{\mathsf{KeyGen}}$ on gid_i.

$\underline{\mathsf{Game}_{2.i-1.11}}$: is the same as $\mathsf{Game}_{2.i-1.10}$ except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + u_j r_2 \boldsymbol{a}_2 + v_j \cdot \boldsymbol{a}_3)$ for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$, where $r_2 \leftarrow_R \mathbb{Z}_p^*$. This is the reverse of the transition from $\mathsf{Game}_{2.i-1.2}$ and $\mathsf{Game}_{2.i-1.3}$. We have that $\mathsf{Game}_{2.i-1.10} \approx_c \mathsf{Game}_{2.i-1.11}$ from the DDH assumption in $\mathbb{G}_1$, which implies that $\{\boldsymbol{u}, \boldsymbol{r} \leftarrow_R \mathbb{Z}_p^{n-1} : (\llbracket \boldsymbol{u} \rrbracket_1, \llbracket \boldsymbol{r} \rrbracket_1)\} \approx_c$

$\{r_2 \leftarrow_R \mathbb{Z}_p^*, \boldsymbol{u} \leftarrow_R \mathbb{Z}_p^{n-1} : ([\![\boldsymbol{u}]\!]_1, [\![r_2 \cdot \boldsymbol{u}]\!]_1)\}$[6]. This first distribution corresponds to $\mathsf{Game}_{2.i-1.10}$, whereas the last distribution corresponds to $\mathsf{Game}_{2.i-1.11}$.

$\underline{\mathsf{Game}_{2.i-1.12}}$: is the same as $\mathsf{Game}_{2.i-1.11}$ except that the output of the hash function on the i'th queried global identifier, which we denote by gid_i, is computed as follows: $H(\mathsf{gid}_i) = [\![\boldsymbol{z}_{\mathsf{gid}_i} + \boldsymbol{a}_3^*]\!]_2$ where $\boldsymbol{z}_{\mathsf{gid}_i} \leftarrow_R \mathsf{Span}(\boldsymbol{a}_1^*)$, as opposed to uniformly random over $\mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_2^*)$ in $\mathsf{Game}_{2.i-1.11}$. This is the reverse of the transition from $\mathsf{Game}_{2.i-1.1}$ and $\mathsf{Game}_{2.i-1.2}$. We have $\mathsf{Game}_{2.i-1.1} \approx_c \mathsf{Game}_{2.i-1.2}$ by **Assumption 2**.

$\underline{\mathsf{Game}_{2.i}}$: is the same as $\mathsf{Game}_{2.i-1.12}$, except that the challenge ciphertext uses the vectors $\boldsymbol{x}_j = (s_j, u_j r_1 \boldsymbol{a}_1 + v_j \boldsymbol{a}_3)$ for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$. That is, we remove the component $u_j r_2 \boldsymbol{a}_2$. We argue that $\mathsf{Game}_{2.i-1.12} \approx_c \mathsf{Game}_{2.i}$ thanks to the super-selective security of $\mathit{\Gamma}$ and $\mathit{\Sigma}$. Indeed, for all $j \in [\ell]$ such that $\theta(j) \in \mathcal{S}_{\mathsf{hon}}$ and all queried gid, we have $\boldsymbol{z}_{\mathsf{gid}} \in \mathsf{Span}(\boldsymbol{a}_1^*, \boldsymbol{a}_3^*)$, thus $(s_j, u_j r_1 \boldsymbol{a}_1 + u_j r_2 \boldsymbol{a}_2 + v_j \cdot \boldsymbol{a}_3)^\top (1, \boldsymbol{z}_{\mathsf{gid}}) = (s_j, u_j r_1 \boldsymbol{a}_1 + v_j \cdot \boldsymbol{a}_3)^\top (1, \boldsymbol{z}_{\mathsf{gid}})$, just as in game $\mathsf{Game}_{2.i}$, since $\boldsymbol{a}_2^\top \boldsymbol{a}_1^* = \boldsymbol{a}_2^\top \boldsymbol{a}_3^* = 0$. □

References

ACGU20. Michel Abdalla, Dario Catalano, Romain Gay, and Bogdan Ursu. Inner-product functional encryption with fine-grained access control. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 467–497. Springer, Heidelberg, December 2020. 5, 16, 22

AKW18. Shashank Agrawal, Venkata Koppula, and Brent Waters. Impossibility of simulation secure functional encryption even with random oracles. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 659–688. Springer, Heidelberg, November 2018. 7

ALS16. Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 333–362. Springer, Heidelberg, August 2016. 7

AYY22. Shweta Agrawal, Anshu Yadav, and Shota Yamada. Multi-input attribute based encryption and predicate encryption. In *Crypto*, 2022. 7

Bei96. Amos Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. Ph.D., Technion - Israel Institute of Technology, 1996. 9

BSW11. Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Heidelberg, March 2011. 5, 11

CC09. Melissa Chase and Sherman S. M. Chow. Improving privacy and security in multi-authority attribute-based encryption. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *ACM CCS 2009*, pages 121–130. ACM Press, November 2009. 2

[6] Again, strictly speaking, the DDH as per Definition 3 is stated with $r_2 \leftarrow_R \mathbb{Z}_p$, not $r_2 \leftarrow_R \mathbb{Z}_p^*$ but as we argued above, this makes no difference since the two distributions are within negligible statistical distance.

Cha07. Melissa Chase. Multi-authority attribute based encryption. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 515–534. Springer, Heidelberg, February 2007. 2

CS02. Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 45–64. Springer, Heidelberg, April / May 2002. 7

DKW21a. Pratish Datta, Ilan Komargodski, and Brent Waters. Decentralized multi-authority ABE for DNFs from LWE. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 177–209. Springer, Heidelberg, October 2021. 7

DKW21b. Pratish Datta, Ilan Komargodski, and Brent Waters. Decentralized multi-authority abe for ncˆ 1 from computational-bdh. *Cryptology ePrint Archive*, 2021. 8

DP19. Edouard Dufour Sans and David Pointcheval. Unbounded inner-product functional encryption with succinct keys. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 19*, volume 11464 of *LNCS*, pages 426–441. Springer, Heidelberg, June 2019. 16

GPSW06. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309. 1

Kim19. Sam Kim. Multi-authority attribute-based encryption from lwe in the ot model. *IACR Cryptol. ePrint Arch.*, 2019:280, 2019. 7

KW93. M. Karchmer and A. Wigderson. On span programs. In *Structure in Complexity Theory Conference, 1993., Proceedings of the Eighth Annual*, pages 102–111, May 1993. 9, 10

LCLS08. Huang Lin, Zhenfu Cao, Xiaohui Liang, and Jun Shao. Secure threshold multi authority attribute based encryption without a central authority. In Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, *INDOCRYPT 2008*, volume 5365 of *LNCS*, pages 426–436. Springer, Heidelberg, December 2008. 2

Lew12. Allison B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 318–335. Springer, Heidelberg, April 2012. 5, 23, 24

LW11. Allison B. Lewko and Brent Waters. Decentralizing attribute-based encryption. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 568–588. Springer, Heidelberg, May 2011. 2, 4, 12

MJ18. Yan Michalevsky and Marc Joye. Decentralized policy-hiding ABE with receiver privacy. In Javier López, Jianying Zhou, and Miguel Soriano, editors, *ESORICS 2018, Part II*, volume 11099 of *LNCS*, pages 548–567. Springer, Heidelberg, September 2018. 7

MKE08. Sascha Müller, Stefan Katzenbeisser, and Claudia Eckert. Distributed attribute-based encryption. In *International Conference on Information Security and Cryptology*, pages 20–36. Springer, 2008. 2

OSW07. Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *ACM CCS 2007*, pages 195–203. ACM Press, October 2007. 3, 9, 21

OT09. Tatsuaki Okamoto and Katsuyuki Takashima. Hierarchical predicate encryption for inner-products. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 214–231. Springer, Heidelberg, December 2009. 5

OT13. Tatsuaki Okamoto and Katsuyuki Takashima. Decentralized attribute-based signatures. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 125–142. Springer, Heidelberg, February / March 2013. 2, 4, 7

Rog15. Phillip Rogaway. The moral character of cryptographic work. *IACR Cryptol. ePrint Arch.*, 2015:1162, 2015. 1

RW15. Yannis Rouselakis and Brent Waters. Efficient statically-secure large-universe multi-authority attribute-based encryption. In Rainer Böhme and Tatsuaki Okamoto, editors, *FC 2015*, volume 8975 of *LNCS*, pages 315–332. Springer, Heidelberg, January 2015. 2, 4

SW05. Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Heidelberg, May 2005. 1

TT18. Junichi Tomida and Katsuyuki Takashima. Unbounded inner product functional encryption from bilinear maps. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 609–639. Springer, Heidelberg, December 2018. 16

WFL19. Zhedong Wang, Xiong Fan, and Feng-Hao Liu. FE for inner products and its application to decentralized ABE. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 97–127. Springer, Heidelberg, April 2019. 8

WWW22. Brent Waters, Hoeteck Wee, and David J Wu. Multi-authority abe from lattices without random oracles. In *Theory of Cryptography Conference*. Springer, 2022. 7