Decentralized Multi-authority ABE for DNFs from LWE

Pratish Datta[1(✉)], Ilan Komargodski[1,2], and Brent Waters[1,3]

[1] NTT Research, Sunnyvale, CA 94085, USA
pratish.datta@ntt-research.com
[2] Hebrew University of Jerusalem, 91904 Jerusalem, Israel
ilank@cs.huji.ac.il
[3] University of Texas at Austin, Austin, TX 78712, USA
bwaters@cs.utexas.edu

Abstract. We construct the first decentralized multi-authority attribute-based encryption (MA-ABE) scheme for a non-trivial class of access policies whose security is based (in the random oracle model) solely on the Learning With Errors (LWE) assumption. The supported access policies are ones described by DNF formulas. All previous constructions of MA-ABE schemes supporting any non-trivial class of access policies were proven secure (in the random oracle model) assuming various assumptions on bilinear maps.

In our system, any party can become an authority and there is no requirement for any global coordination other than the creation of an initial set of common reference parameters. A party can simply act as a standard ABE authority by creating a public key and issuing private keys to different users that reflect their attributes. A user can encrypt data in terms of any DNF formulas over attributes issued from any chosen set of authorities. Finally, our system does not require any central authority. In terms of efficiency, when instantiating the scheme with a global bound s on the size of access policies, the sizes of public keys, secret keys, and ciphertexts, all grow with s.

Technically, we develop new tools for building ciphertext-policy ABE (CP-ABE) schemes using LWE. Along the way, we construct the first provably secure CP-ABE scheme supporting access policies in NC^1 under the LWE assumption that avoids the generic universal-circuit-based key-policy to ciphertext-policy transformation. In particular, our construction relies on linear secret sharing schemes with new properties and in some sense is more similar to CP-ABE schemes that rely on bilinear maps. While our CP-ABE construction is not more efficient than existing ones, it is conceptually intriguing and further we show how to extend it to get the MA-ABE scheme described above.

1 Introduction

Attribute-based encryption (ABE) is a generalization of traditional public-key encryption [26] that offers fine-grained access control over encrypted data based on the credentials (or attributes) of the recipients. ABE comes in two avatars:

A. Canteaut and F.-X. Standaert (Eds.): EUROCRYPT 2021, LNCS 12696, pp. 177–209, 2021.
https://doi.org/10.1007/978-3-030-77870-5_7

ciphertext-policy and *key-policy*. In a ciphertext-policy ABE (CP-ABE), as the name suggests, ciphertexts are associated with access policies and keys are associated with attributes. In a key-policy ABE (KP-ABE), the roles of the attribute sets and the access policies are swapped, i.e., ciphertexts are associated with attributes and keys are associated with access policies. In both cases, decryption is possible *only when* the attributes satisfy the access policy.

Since its inception by Sahai and Waters, and Goyal et al. [37,54], ABE has become a fundamental cryptographic primitive with a long list of potential applications. Therefore, designing ABE schemes has received tremendous attention by the cryptographic community resulting in a long sequence of works achieving various trade-offs between expressiveness, efficiency, security, and underlying assumptions [5,8,11,13,15,18,19,23,24,27,31–33,36,40,41,43,50,56,58].

Most of the aforementioned works base their security on cryptographic assumptions related to bilinear maps. It is very natural to seek for constructions based on other assumptions. First, this is important from a conceptual perspective as not only more constructions increase our confidence in the existence of a scheme, but constructions using different assumptions often require new techniques which in turn improves our understanding of the primitive. Second, this is important in light of the known attacks on group-based constructions by quantum computers [55]. Within this general goal, we currently have a handful of ABE schemes (that go beyond Identity-Based Encryption) [4,5,15,16,18,19,33,34,56] which avoid bilinear maps as their underlying building blocks.

All of these works derive their security from the hardness of the *learning with errors* (LWE) problem, which is currently also believed to be hard against quantum computers [29,46,47,51,52]. However, one striking fact is that all existing LWE-based ABE schemes (mentioned above) are designed in the key-policy setting. To date, the natural dual problem of constructing CP-ABE schemes based on the LWE assumption is essentially completely open.

The only known way to realize an LWE-based CP-ABE scheme is to convert either of the circuit-based KP-ABE schemes of [15,19,33] into a CP-ABE scheme by using a universal circuit to represent an access policy as an attribute and an attribute set as a circuit. However, this transformation will inherently result with a CP-ABE for a restricted class of access policies and with parameters that are far from ideal. Concretely, for any polynomials s, d in the security parameter, it allows to construct a CP-ABE for access policies with circuits of size s and depth d. Moreover, the size of a ciphertext generated with respect to some access policy f will be $|f| \cdot \mathsf{poly}(\lambda, s, d)$ (no matter what KP-ABE we start off with). That is, even if an f being encrypted has a very small circuit, the CP-ABE ciphertext would scale with the *worst-case* bounds s, d.

Open Problem 1: *Improve* (*even modestly*) *upon the universal-circuit based* CP-ABE *construction described above while assuming only* LWE.

There have been few recent exciting attempts towards this problem [6–8,18]. The works of [6,8,18] attempt go all the way and construct a *succinct* CP-ABE, where there is no global size bound s and ciphertexts and keys are of size independent of s. The works [6,8], rely on LWE as well as on bilinear groups (either

generic [8] or a particular knowledge assumption [6]). The work [18] lacks a security proof. Most recently, [7] constructed a CP-ABE scheme based on LWE that still requires a universal circuit size bound but the sizes of ciphertexts and keys are independent of it.

Multi-authority Attribute-Based Encryption: In an ABE scheme, keys can only be generated and issued by a central authority. A natural extension of this notion, introduced by Chase [21] and termed multi-authority ABE (MA-ABE), allows multiple parties to play the role of an authority. In an MA-ABE, there are multiple authorities which control different attributes and each of them can issue secret keys to users possessing attributes under their control without any interaction with the other authorities in the system. Specifically, given a ciphertext generated with respect to some access policy, a user possessing a set of attributes satisfying the access policy can decrypt the ciphertext by pulling the individual secret keys it obtained from the various authorities controlling those attributes. The security requires collusion resistance against unauthorized users with the important difference that now some of the attribute authorities may be corrupted and therefore may collude with the adversarial users.

To date, there are only a few works which have dealt with the problem of constructing MA-ABE schemes. After few initial attempts [21,22,44,48,49] that had various limitations, Lewko and Waters [42] were able to design a truly decentralized MA-ABE scheme in which any party can become an authority and there is no requirement for any global coordination other than the creation of an initial trusted setup. In their scheme, a party can simply act as an authority by publishing a public key of its own and issuing private keys to different users that reflect their attributes. Different authorities need not even be aware of each other and they can join the system at any point of time. There is also no bound on the number of attribute authorities that can ever come into play during the lifetime of the system. Their scheme supports access policies computable by NC^1 circuits and their security is proven in the random oracle model and further relies on assumptions on bilinear groups (similarly to all previous MA-ABE constructions). Rouselakis and Waters [53] provided further efficiency improvements over [42], albeit they rely, in addition to a random oracle, on a non-standard q-type assumption.

Open Problem 2: *Is there a truly decentralized* MA-ABE *for some non-trivial class of access policies assuming hardness of* LWE (*and in the random oracle model*)*?*

There has been few recent attempts at this problem as well [39,57]. Both constructions [39,57] assume a central authority which generates the public and secret keys for all the attribute authorities in the system. Thus all authorities that will ever exist in the system are forever fixed once setup is complete which runs counter to the truly decentralized spirit of [42]. Additionally, both schemes guarantee security only against a bounded collusion of parties. In fact, the scheme of Kim [39] is built in a new model, called the "OT model", which is incapable of

handling even bounded collusion.[1] In this sense, both constructions suffer from related limitations to the early MA-ABE constructions [21,22,44,48,49] describe above. The differences between the two constructions are that the scheme of Wang et al. [57] supports NC^1 access policies, while the scheme due to Kim [39] support arbitrary bounded depth circuits.

1.1 Our Contributions

In this paper, we make progress with respect to Open Problem 2, stated above. We construct a new MA-ABE scheme supporting an unbounded number of attribute authorities for access policies captured by DNF formulas. Our scheme is proven secure in the random oracle model and relies on the hardness of the LWE problem.

Theorem 1.1 (Informal): *There exist a decentralized* MA-ABE *scheme for access policies captured by* DNF *formulas under the* LWE *assumption. Our scheme is (statically) secure against an arbitrary collusion of parties in the random oracle model and assuming the* LWE *assumption with subexponential modulus-to-noise ratio.*

Similarly to [42,53], in our MA-ABE scheme, any party can become an authority at any point of time and there is no bound on the number of attribute authorities that can join the system or need for any global coordination other than the creation of an initial set of common reference parameters created during a trusted setup. We prove the security of our MA-ABE scheme in the static security model introduced by Rouselakis and Waters [53] where all of the ciphertexts, secret keys, and corruption queries must be issued by the adversary before the public key of any attribute authority is published.

Towards obtaining Theorem 1.1, we make conceptual contribution towards Open Problem 1. We present the first provably secure direct CP-ABE construction which avoids the generic universal-circuit-based key-policy to ciphertext-policy transformation. In particular, our approach deviates from all previous LWE-based expressive ABE constructions [5,15,18,19,33,34,56] that are in turn based on techniques inspired by fully homomorphic encryption [28,30]. In contrast, our CP-ABE is based on useful properties of linear secret sharing schemes

[1] All previous multi-authority ABE schemes were designed in the so called global identifier (GID) model where each user in the system is identified by a unique global identity string $\mathsf{GID} \in \{0,1\}^*$. The global identity of a user remains fixed for the entire lifetime of the system and users have no freedom to choose their global identities. Kim [39] introduced a drastically relaxed model, the so called "OT model", where each user can self-generate some key-request string and produce it to the attribute authorities while requesting secret keys. To briefly see why this model fails to guarantee collusion resistance, imagine that there are two users A who has attribute u and B who has attribute v. Suppose there is a ciphertext encrypting to the policy "u AND v". User A and B can collude to decrypt it. Morally, the issue is that user A can go with the authority for attribute u and produce a key with identity George. User B can then present the same identity to the authority for attribute v. Then they can combine their keys.

and can be viewed as the LWE analog of the CP-ABE scheme of Waters [58] which relies on the decisional bilinear Diffie-Hellman assumption.

Theorem 1.2 (Informal): *There exist a* CP-ABE *scheme supporting all access policies in* NC^1. *The scheme is selectively secure assuming the* LWE *assumption with subexponential modulus-to-noise ratio.*

Our CP-ABE scheme achieves the standard selective security where the adversary must disclose its ciphertext query before the master public key is published but is allowed to make secret key queries adaptively throughout the security experiment. Again, Theorem 1.2 does not improve upon previously known constructions in any parameter. It is in fact worse in several senses: it only supports NC^1 access policies, its efficiency is worse, and it requires the LWE assumption to hold with subexponential modulus-to-noise ratio. However, the new construction is interesting not only because we show how to generalize it to get the new MA-ABE scheme from Theorem 1.1, but also because we introduce a conceptually new approach and develop several interesting tools and proof techniques.

One highlight is that we distill a set of properties of linear secret sharing schemes (LSSS) which makes them compatible with LWE-based constructions. Specifically, we instantiate both of our CP-ABE and MA-ABE schemes with such LSSS schemes. In the security model of CP-ABE we are able to construct such a compatible LSSS for all NC^1 while in the (much harder) security model of MA-ABE we are only able to get such a scheme for DNFs. The properties are:

- **Small reconstruction coefficients:** The reconstruction coefficients of the LSSS must be small, say $\{0, 1\}$. This property of LSSS secret sharing schemes was recently formally defined by [14]. They observed that a well-known construction by Lewko and Waters [42] actually results with an LSSS with this property for all access structures in NC^1.
- **Linear independence for unauthorized rows:** This property says that rows of the share generating matrix that correspond to an unauthorized set of parties are linearly independent. Agrawal et al. [1] recently observed that the aforementioned construction by Lewko and Waters [42], when applied on DNF access structures, results with a share generating matrix that has this property as well.

Both of our constructions, the CP-ABE as well as the MA-ABE, are actually designed to work with any access structure that has an LSSS with the above two properties.

Theorem 1.3 (Informal): *Consider a class of access policies* $\mathbb{P}$ *that has an associated* LSSS *with the above two properties. Then, there exists a* CP-ABE *and an* MA-ABE *supporting access policies from the class* $\mathbb{P}$. *Both schemes are secure assuming the* LWE *assumption with subexponential modulus-to-noise ratio and the* MA-ABE *scheme also requires a random oracle.*

To obtain Theorem 1.2 we design a new (non-monotone) LSSS for all NC^1 that has the above two properties. This is summarized in the following theorem.

Theorem 1.4 (Informal): *There exists a non-monotone* LSSS *scheme for all* NC^1 *circuits satisfying the* small reconstruction coefficients *and* linear independence for unauthorized rows *properties.*

By non-monotone, we mean that an attribute and its negation are treated separately (both having corresponding shares) and it is implicitly assumed that the attacker will never see shares corresponding to both the positive and the negative instances of the same attribute. This can be enforced in case of CP-ABE due to its centralized nature and this when combined with Theorem 1.3 implies Theorem 1.2. However, in MA-ABE attackers can get hold of the master secret key of any attribute authority and generate secret keys corresponding to both the attribute under control and its negation, and so non-monotone LSSS does not seem to suffice. We therefore settle for the (monotone) LSSS scheme for DNFs to obtain Theorem 1.1 (see further discussion in Sect. 2.3 below and [25, Remark 6.1] in the full version).

Boyen's [16] scheme: In TCC 2013 Boyen [16] suggested a lattice-based KP-ABE scheme for NC^1. While being conceptually similar to analogous constructions from the bilinear-maps LSSS-based schemes, soon after the publication a flaw was found and a recent work of Agrawal et al. [1] shows an explicit attack. The attack of [1] is based on identifying a subset of attributes which correspond to rows of the policy matrix that non-trivially span the $\mathbf{0}$ vector (i.e., linearly dependent rows). To rescue Boyen's construction, Agrawal et al. [1] suggest to use an LSSS which has the linear independence of unauthorized rows property (they call it an *admissible LSSS*), however, they fail to obtain such a scheme for any class larger than DNFs. Our non-monotone LSSS scheme for NC^1 (Theorem 1.4) can be used to resurrect the KP-ABE scheme of Boyen [16]. Although this does not imply any new result (as other constructions of KP-ABE for all polynomial-size circuits have since been discovered [15,19,33]), we believe that this is an important conceptual contribution.

Paper Organization: In Sect. 2 we provide a high-level overview of our techniques. Prerequisites on lattices and LWE are provided in Sect. 3. In Sect. 4 we give our construction of the new non-monotone LSSS for all NC^1 with the linear independence property. In Sect. 5 we give the construction of our CP-ABE scheme and prove its correctness. The proof of security is provided in the full version [25]. In Sect. 6 we give the construction of our MA-ABE scheme. The proofs of correctness and security are again deferred to the full version [25]. We further omit the formal syntax and security definitions of CP-ABE and MA-ABE in this version. Those can be found in the full version [25].

2 Technical Overview

In this section we provide a high level overview of our main ideas and techniques. In a very high level, our CP-ABE construction is composed of two main conceptual ideas:

1. *A linear non-monotone secret sharing scheme with small reconstruction coefficients and a* linear independence *guarantee*: We design a new linear non-monotone secret sharing scheme for all access structures that can be described by a Boolean *formula*, namely NC^1 access structures. The new secret sharing scheme possesses two properties which turns out to be key for our correctness and security proof. The first property states that it is possible to reconstruct a shared secret using only coefficients that come from $\{0,1\}$. An LSSS with this property is called $\{0,1\}$-LSSS [14]. The second property, called the linear independence property, says that the shares held by any unauthorized set, not only are independent of the secret, but are also linearly independent among each other. We give an overview of the new construction in Sect. 2.1.
2. *An* LWE*-based direct construction of* CP-ABE: We show how to leverage any $\{0,1\}$-LSSS with the above extra property to get a CP-ABE scheme. Conceptually, to some extent the construction can be viewed as a "translation" of Waters' [58, Section 6] construction of a CP-ABE scheme under the Decisional Bilinear Diffie-Hellman (DBDH) Assumption into the LWE regime. However, since we are basing the construction of the LWE assumption, the details and implementation are completely different and much more involved. We will give an overview of this part in Sect. 2.2.

Combining the two parts, we obtain a CP-ABE scheme for all NC^1 assuming the LWE assumption. The CP-ABE scheme we design is already amenable for extension to the multi-authority setting. We briefly discuss the main idea in the extension to MA-ABE in Sect. 2.3.

2.1 The New Linear Secret Sharing Scheme

Our goal is to construct a linear secret sharing scheme with $\{0,1\}$ reconstruction coefficients where the shares of unauthorized parties are linearly independent. Recall first that an access structure f is a partition of the universe of possible subsets of n parties into two sets, one is called *authorized* and its complement is called *unauthorized*. The partition is monotone in the sense that if some subset of parties is unauthorized, one can make it authorized only by adding more parties to it. A secret sharing scheme is a method by which it is possible to "split" a given secret into "shares" and distributes them among parties so that authorized subsets would be able to jointly recover the secret while others would not. Linear secret sharing schemes (LSSS) [38] are a subset of all possible schemes where there is an additional structural guarantee about the reconstruction procedure: For an authorized subset of parties to reconstruct the secret, all that is needed is to compute a *linear* function over its shares.

Every linear secret sharing scheme can be described by a share generating matrix. This is a matrix $\boldsymbol{M} \in \mathbb{Z}_q^{\ell \times d}$ where each row is associated to some party. A set of parties is qualified if and only if when we restrict $\boldsymbol{M}$ to rows of this set, we get a subspace that spans the vector $(1,0,\ldots,0)$. For a secret $z \in \mathbb{Z}_q$, computing $\boldsymbol{M} \cdot \boldsymbol{v}^\top$, where $\boldsymbol{v} \in \mathbb{Z}_q^d$ is a vector whose first entry is z and the rest are uniformly random, gives a vector of ℓ shares of the secret z. Here, we need a more specialized share generating matrix with an additional property.

Specifically, we need that for any unauthorized set of parties, restricting $\boldsymbol{M}$ to those rows, results with a set of linearly independent vectors. We construct such a share generating matrix for access structure given as a Boolean formula.

To see the challenge, it is useful to recall the standard construction of a share generating matrix for Boolean formulas, as adapted from the secret sharing scheme of [12] by Lewko and Waters [42, Appendix G]. Given a Boolean formula, the share generating matrix is constructed by labeling the wires of the formula from the root to the leaves. The labels of the leaves will form the rows of the share generating matrix. We first label the root node of the tree with the vector (1) (a vector of length 1). Then, we go down the levels of the tree one by one, labeling each node with a vector determined by the vector assigned to its parent node. Throughout the process, we maintain a global counter variable c which is initialized to 1. Consider a gate g with output wire w whose label is $\boldsymbol{w}$ and two input wires u, v. If g is an OR gate, we associate with u the label $\boldsymbol{u} = \boldsymbol{w}$ and with v the label $\boldsymbol{v} = \boldsymbol{w}$ (and do not change c). If g is an AND gate, we associate with u the label $\boldsymbol{u} = \boldsymbol{w}\|1$ and associate with v the label $\boldsymbol{v} = \boldsymbol{0}\| - 1$, where $\boldsymbol{0}$ denoted a length c vector of 0s. We now increment the value of c by 1. Finally all vectors are padded with 0s in the end to the length of the longest one.

Let us mention that this scheme already has several appealing properties. First, the entries of the share generating matrix are from $\{-1, 0, 1\}$. Moreover, it is already a $\{0, 1\}$-LSSS, namely, when reconstructing a secret using the shares corresponding to an authorized set, the coefficients used are only from $\{0, 1\}$. Nevertheless, a property that we need yet the above construction does not satisfy is linear independence. Consider, for instance, the formula $(A \vee B) \wedge C$. Here, an adversary controlling A and B cannot recover the secret, yet the rows corresponding to A and B in the share generating matrix are identical and thereby linearly dependent. The more intuitive way to see the problem is that during the reconstruction process, since we are dealing with an OR gate, we can choose to continue "either from the left or from the right" and in both cases we will see the same computation. Nevertheless, it is not hard to verify that when considering only DNF formulas, this construction already results with linearly independent rows for unqualified sets.

We next describe our new secret sharing scheme and argue that the rows corresponding to any unauthorized set are linearly independent. We make our task a little bit easier by allowing every wire in the formula have two associated labels. (This is why our scheme is a non-monotone LSSS.) The first is for "satisfying" the wire, i.e., the 1-label, and the other is for not satisfying it, i.e., the 0-label. (Whereas above we only had a label for satisfying the wire and hence it is a monotone LSSS.) Our procedure is similar to the one above in the sense that it also labels wires from the root to the leaves and the leaf labels form the rows of the share generating matrix. Since we have two labels per wire, we first label the root node of the tree with the vector (1,0) and (0,1). Our global counter c is initialized to 2.

Consider a gate g with output wire w whose labels are $\boldsymbol{w}_1, \boldsymbol{w}_0$, and two input wires u, v. We associate with u the labels $\boldsymbol{u}_1, \boldsymbol{u}_0$ and with v the label $\boldsymbol{v}_1, \boldsymbol{v}_0$. If

g is an AND gate, we set

$$\boldsymbol{u}_1 = \mathbf{0}\|1, \quad \boldsymbol{u}_0 = \boldsymbol{w}_0, \quad \boldsymbol{v}_1 = \boldsymbol{w}_1\| - 1, \quad \boldsymbol{v}_0 = \boldsymbol{w}_0\| - 1$$

If g is an OR gate, we set

$$\boldsymbol{u}_1 = \boldsymbol{w}_1, \quad \boldsymbol{u}_0 = \mathbf{0}\|1, \quad \boldsymbol{v}_1 = \boldsymbol{w}_1\| - 1, \quad \boldsymbol{v}_0 = \boldsymbol{w}_0\| - 1$$

We increment the value of c by 1 and pad all vectors with 0s in the end to be of size c.

Correctness and security of the construction (which can be proven by induction) say that for every wire in the formula, if it can be successfully satisfied, then there is a linear combination to recover the 1-label of that wire but not the 0-label. Analogously, if it cannot be satisfied, then there is also a linear combination to recover the 0-label of that wire but not the 1-label. Also, it is not hard to verify that, as with the previous construction, the matrix contains only values from $\{-1, 0, 1\}$ and the reconstruction coefficients needed to recover the secret for an authorized set are from $\{0, 1\}$.

For the new linear independent property, let us focus for now on a single gate g and assume that it is an OR gate. Observe that $\boldsymbol{w}_1$ can only be reconstructed using either $\boldsymbol{u}_1$ or using $\boldsymbol{u}_0 + \boldsymbol{v}_1$. As opposed to the "attack" we suggested before, now to continue the computation in the reconstruction phase, there is only one valid way, depending on the available shares. To see this more precisely, one needs to consider the 4 possible cases: (1) u, v are satisfied, (2) u is satisfied but v is not, (3) u is not satisfied but v is, and (4) both u, v are unsatisfied. Checking each case separately one can get convinced that there is exactly one way to compute the corresponding label of the output wire. An analogous case analysis can be done also for the case where g is an AND gate. This idea can be generalized and formalized to show that the vectors held by an attacker who controls an unauthorized must be linearly independent.

2.2 The CP-ABE Scheme

Here we describe our CP-ABE scheme. This serves as a warm up for our full MA-ABE scheme and includes most of the technical ideas. We discuss briefly the additional technicalities that arise in the multi-authority setting in Sect. 2.3 below. Note that the problem of constructing CP-ABE schemes directly has traditionally been much more challenging compared to its KP-ABE counterpart. Let us highlight two challenges:

- The first challenge is of course to prevent collusion attacks by users, that is, to somehow "bind" the key components of a particular user corresponding to the various attributes it possesses so that those key components cannot be combined with the key components possessed by other users.
- The second and more serious challenge is (in the selective model) how to embed a complex access policy in a short number of parameters.

In order to prove selective security, the standard strategy is to follow a "partitioning" technique where the reduction algorithm sets up the master public key such that it knows all the secret keys that it needs to give out, yet it cannot give out secret keys that can trivially decrypt the challenge ciphertext. In the context of KP-ABE, the challenge ciphertext is associated with an attribute set and therefore the public parameters for each attribute can be simply treated differently depending whether it is in the challenge attribute set or not. In CP-ABE, the situation is much more complicated as ciphertexts are associated with access policies which essentially encode a huge (maybe exponential size) set of authorized subsets of attributes. Consequently, there is no simple "on or off" method of programming this information into the master public key. While techniques have eventually been developed to overcome this challenge in the bilinear map world, devising the LWE analogs has remained elusive. One of the main technical contributions of our paper is a method for directly embedding an LSSS access policy into the master public key within the LWE-based framework in our reduction.

For concreteness, in what follows we assume that the LSSS access policy used in our CP-ABE scheme was generated using our transformation described above. Moreover, we assume that there is a public bound $s_{\max}$ on the number of columns in the matrix (which translates to a bound on the size of the Boolean formula while using our Boolean formula LSSS transformations above). We further assume that the row labeling function is injective, i.e., each attribute corresponds to exactly one row. In the precise description of the scheme we use several different noise distributions with varying parameters. Some of them are used to realize the standard noise smudging technique at various steps of the security proof. In order to keep the exposition simple, we will ignore such noise smudging and just use a single noise distribution, denoted noise. By default, vectors are thought of as row vectors.

Setup: For each attribute u in the system, sample $\boldsymbol{A}_u \in \mathbb{Z}_q^{n \times m}$ together a trapdoor $\boldsymbol{T}_{\boldsymbol{A}_u}$, and another uniformly random matrix $\boldsymbol{H}_u \leftarrow \mathbb{Z}_q^{n \times m}$. Additionally sample $\boldsymbol{y} \leftarrow \mathbb{Z}_q^n$. Output

$$\mathsf{PK} = (\boldsymbol{y}, \{\boldsymbol{A}_u\}, \{\boldsymbol{H}_u\}), \qquad \mathsf{SK} = \{\boldsymbol{T}_{\boldsymbol{A}_u}\}$$

Key Generation for attribute set U: Let $\hat{\boldsymbol{t}} \leftarrow \mathsf{noise}^{m-1}$ and $\boldsymbol{t} = (1, \hat{\boldsymbol{t}}) \in \mathbb{Z}^m$. This vector $\boldsymbol{t}$ will intuitively serve as the linchpin that will tie together all the secret key components of a specific user. For each attribute $u \in U$, using $\boldsymbol{T}_{\boldsymbol{A}_u}$, sample a short vector $\tilde{\boldsymbol{k}}_u$ such that $\boldsymbol{A}_u \tilde{\boldsymbol{k}}_u^\top = \boldsymbol{H}_u \boldsymbol{t}^\top$ and output

$$\mathsf{SK} = (\{\tilde{\boldsymbol{k}}_u\}, \boldsymbol{t})$$

Encryption of $\mathsf{msg} \in \{0,1\}$ given matrix $\boldsymbol{M}$: Assume that ρ is a function that maps between row indices of $\boldsymbol{M}$ and attributes, that is, $\rho(i)$ is the attribute associated with the ith row in $\boldsymbol{M}$. The procedure samples $\boldsymbol{s} \leftarrow \mathbb{Z}_q^n$ and

$\boldsymbol{v}_2, \ldots, \boldsymbol{v}_{s_{\max}} \leftarrow \mathbb{Z}_q^m$ and computes

$$\boldsymbol{c}_i = \boldsymbol{s}\boldsymbol{A}_{\rho(i)} + \mathsf{noise}$$
$$\hat{\boldsymbol{c}}_i = M_{i,1}(\boldsymbol{s}\boldsymbol{y}^\top, \overbrace{0, \ldots, 0}^{m-1}) + \left[\sum_{j \in \{2,\ldots,s_{\max}\}} M_{i,j}\boldsymbol{v}_j\right] - \boldsymbol{s}\boldsymbol{H}_{\rho(i)} + \mathsf{noise}$$

and outputs the ciphertext

$$\mathsf{CT} = \left(\{\boldsymbol{c}_i\}_{i\in[\ell]}, \{\hat{\boldsymbol{c}}_i\}_{i\in[\ell]}, C = \mathsf{MSB}(\boldsymbol{s}\boldsymbol{y}^\top) \oplus \mathsf{msg}\right).$$

Decryption: Assume that the available attributes are qualified to decrypt. Let I be the set of row indices corresponding to the available attributes and let $\{w_i\}_{i\in I} \in \{0,1\} \subset \mathbb{Z}_q$ be the reconstruction coefficients. For each $i \in I$, let $\rho(i)$ be the attribute associated with the ith row. The procedure computes

$$K' = \sum_{i\in I} w_i \left(\boldsymbol{c}_i \tilde{\boldsymbol{k}}_{\rho(i)}^\top + \hat{\boldsymbol{c}}_i \boldsymbol{t}^\top\right)$$

and outputs

$$\mathsf{msg}' = C \oplus \mathsf{MSB}(K').$$

Correctness

Consider a ciphertext CT w.r.t some matrix $\boldsymbol{M}$ and a key for a set of attributes U that satisfies $\boldsymbol{M}$. By construction it is enough to show that $\mathsf{MSB}(K') = \mathsf{MSB}(\boldsymbol{s}\boldsymbol{y}^\top)$ with all but negligible probability. Here, for simplicity, we shall ignore small noise-like terms. Expanding $\{\boldsymbol{c}_i\}_{i\in I}$ and $\{\hat{\boldsymbol{c}}_i\}_{i\in I}$, we get

$$\begin{aligned} K' \approx & \sum_{i\in I} w_i \boldsymbol{s}\boldsymbol{A}_{\rho(i)}\tilde{\boldsymbol{k}}_{\rho(i)}^\top + \sum_{i\in I} w_i M_{i,1}(\boldsymbol{s}\boldsymbol{y}^\top, 0, \ldots, 0)\boldsymbol{t}^\top \\ & + \sum_{i\in I, j\in\{2,\ldots,s_{\max}\}} w_i M_{i,j}\boldsymbol{v}_j\boldsymbol{t}^\top - \sum_{i\in I} w_i \boldsymbol{s}\boldsymbol{H}_{\rho(i)}\boldsymbol{t}^\top \end{aligned}$$

First, observe that each $w_i \in \{0,1\}$ since the reconstruction coefficients in our secret sharing scheme are guaranteed to be Boolean.

Now, recall that for each $u \in U$, we have $\boldsymbol{A}_u\tilde{\boldsymbol{k}}_u^\top = \boldsymbol{H}_u\boldsymbol{t}^\top$. Therefore, for each $i \in I$, it holds that

$$\boldsymbol{A}_{\rho(i)}\tilde{\boldsymbol{k}}_{\rho(i)}^\top = \boldsymbol{H}_{\rho(i)}\boldsymbol{t}^\top.$$

Hence,

$$\begin{aligned}
K' \approx & \cancel{\sum_{i\in I} w_i \boldsymbol{s}\boldsymbol{H}_{p(i)}\boldsymbol{t}^\top} + \sum_{i\in I} w_i M_{i,1}(\boldsymbol{s}\boldsymbol{y}^\top, 0, \ldots, 0)\boldsymbol{t}^\top \\
& + \sum_{i\in I, j\in\{2,\ldots,s_{\max}\}} w_i M_{i,j}\boldsymbol{v}_j\boldsymbol{t}^\top - \cancel{\sum_{i\in I} w_i \boldsymbol{s}\boldsymbol{H}_{p(i)}\boldsymbol{t}^\top} \\
= & \sum_{i\in I} w_i M_{i,1}(\boldsymbol{s}\boldsymbol{y}^\top, 0, \ldots, 0)\boldsymbol{t}^\top + \sum_{i\in I, j\in\{2,\ldots,s_{\max}\}} w_i M_{i,j}\boldsymbol{v}_j\boldsymbol{t}^\top \\
= & \left(\sum_{i\in I} w_i M_{i,1}\right)(\boldsymbol{s}\boldsymbol{y}^\top, 0, \ldots, 0)\boldsymbol{t}^\top + \sum_{j\in\{2,\ldots,s_{\max}\}} \left(\sum_{i\in I} w_i M_{i,j}\right)\boldsymbol{v}_j\boldsymbol{t}^\top.
\end{aligned}$$

Recall that we have $\sum_{i\in I} w_i M_{i,1} = 1$ while for $1 < j \leq s_{\max}$, it holds that $\sum_{i\in I} w_i M_{i,j} = 0$. Also, recall that $\boldsymbol{t} = (1, \hat{\boldsymbol{t}})$, and hence, $(\boldsymbol{s}\boldsymbol{y}^\top, 0, \ldots, 0)\boldsymbol{t}^\top = \boldsymbol{s}\boldsymbol{y}^\top$. Thus,

$$K' \approx \boldsymbol{s}\boldsymbol{y}^\top.$$

By choosing the noise magnitude carefully, we can make sure that $\mathsf{MSB}(K') = \mathsf{MSB}(\boldsymbol{s}\boldsymbol{y}^\top)$, except with negligible probability.

Security

As mentioned, we prove that our scheme is selectively secure, namely, we require the challenge LSSS policy $(\boldsymbol{M}, \rho)$ to be submitted by the adversary ahead of time before seeing the public parameters. The proof is obtained by a hybrid argument where we start off with the security game played with the real scheme as the first hybrid and end up with a hybrid where the game is played with a scheme where the challenge ciphertext is independent of the underlying message.

In more detail, in the last hybrid we want to get rid of the secret $\boldsymbol{s}$. Recall that $\boldsymbol{s}$ appears in two places: (1) $\boldsymbol{c}_i$ and (2) $\hat{\boldsymbol{c}}_i$. Intuitively, the term $\boldsymbol{c}_i$ looks like an LWE sample and indeed our goal is to use LWE to argue that $\boldsymbol{s}$ is hidden there. The challenge is that to use LWE we need to get rid of the trapdoor $\boldsymbol{T}_{\boldsymbol{A}_u}$ of $\boldsymbol{A}_u$ which is used in the key generation procedure to sample $\tilde{\boldsymbol{k}}_u$. For $\hat{\boldsymbol{c}}_i$, our high level approach is to program $\boldsymbol{H}_u$ in such a way that it will cancel the terms that depend on $\boldsymbol{s}$ in $\hat{\boldsymbol{c}}_i$. However, at the same time $\boldsymbol{H}_u$ is used in the sampling procedure of $\tilde{\boldsymbol{k}}_u$ as well, and so (1) and (2) are actually related and need to be handled together.

We program $\boldsymbol{H}_u$ as follows

$$\boldsymbol{H}_u = M_{\rho^{-1}(u),1}\left[\boldsymbol{y}^\top | \overbrace{\boldsymbol{0}^\top | \cdots | \boldsymbol{0}^\top}^{m-1}\right] + \sum_{j\in\{2,\ldots,s_{\max}\}} M_{\rho^{-1}(u),j}\boldsymbol{B}_j + \boldsymbol{A}_u\boldsymbol{R}_u,$$

where $\boldsymbol{R}_u, \boldsymbol{B}_2, \ldots, \boldsymbol{B}_{s_{\max}}$ are matrices of the appropriate sizes and sampled from some distributions which we shall skip for now. Here we crucially use the fact that the row labeling function ρ is injective to ensure that the above definition

of $\boldsymbol{H}_u$ is unambiguous. One of the purposes of the $\boldsymbol{R}_u$ matrices is to make sure that the programmed $\boldsymbol{H}_u$ is indistinguishable from the original $\boldsymbol{H}_u$. We make use of an extended version of the leftover hash lemma, we call the "leftover hash lemma with trapdoors" (see Lemma 3.4 in the full version [25]), to guarantee this indistinguishability. This programming allows us to embed the challenge access policy into the master public key. Also notice that indeed the first term of $\boldsymbol{H}_u$ cancels out the dependence on $\boldsymbol{s}$ in $\hat{\boldsymbol{c}}_i$.

Let us go back to how the keys look like with this $\boldsymbol{H}_u$. Recall that we chose $\tilde{\boldsymbol{k}}_u$ such that $\boldsymbol{A}_u\tilde{\boldsymbol{k}}_u^\top = \boldsymbol{H}_u\boldsymbol{t}^\top$. Our goal is to sample $\tilde{\boldsymbol{k}}_u$ directly and not through the trapdoor $\boldsymbol{T}_{\boldsymbol{A}_u}$ of $\boldsymbol{A}_u$ so that we can eventually do away with $\boldsymbol{T}_{\boldsymbol{A}_u}$. To this end, we program $\boldsymbol{t}$ so that $\boldsymbol{H}_u\boldsymbol{t}^\top$ is completely random. Note that once $\boldsymbol{H}_u\boldsymbol{t}^\top$ becomes random, we would be able to directly sample $\tilde{\boldsymbol{k}}_u$ via the properties of lattice trapdoors. At a high level for this purpose, we use the $\boldsymbol{B}_j$ matrices, which we actually generate along with trapdoors. Observe that with our programming of the $\boldsymbol{H}_u$ matrices above, we have

$$\boldsymbol{H}_u\boldsymbol{t}^\top = M_{\rho^{-1}(u),1}\left[\boldsymbol{y}^\top|\overbrace{\boldsymbol{0}^\top|\cdots|\boldsymbol{0}^\top}^{m-1}\right]\boldsymbol{t}^\top + \boxed{\sum_{j\in\{2,\ldots,s_{\max}\}} M_{\rho^{-1}(u),j}\boldsymbol{B}_j\boldsymbol{t}^\top} + \boldsymbol{A}_u\boldsymbol{R}_u\boldsymbol{t}^\top.$$

Roughly, $\boldsymbol{H}_u\boldsymbol{t}^\top$ would become uniformly random if we can make the boxed part above uniformly random. We plan to do this by first sampling some uniformly random vector $\boldsymbol{z}_u$ and then solving for $\left\{\boldsymbol{B}_j\boldsymbol{t}^\top\right\}_{j\in\{2,\ldots,s_{\max}\}}$ such that $\sum_{j\in\{2,\ldots,s_{\max}\}} M_{\rho^{-1}(u),j}\left(\boldsymbol{B}_j\boldsymbol{t}^\top\right) = \boldsymbol{z}_u$. Note that once we have a solution for the above system of equations, we can use the trapdoor of the $\boldsymbol{B}_j$ matrices to sample an appropriate $\boldsymbol{t}$ and our goal will be accomplished. It is for solving the above system of linear equations that we use the fact that the corresponding rows of $\boldsymbol{M}$ are linearly independent and so the above system of linear equations is solvable.

2.3 The MA-ABE Scheme

The MA-ABE scheme is a generalization of the above scheme and we avoid repeating the scheme here. Instead, let us go over our main ideas to overcome the technical challenges that prevented getting a collusion resistant decentralized MA-ABE scheme from LWE before this work. First, it is important to understand that a main challenge in CP-ABE constructions is collusion resistance. The standard technique to achieve collusion resistance in the literature is to tie together the different key components representing the different attributes of a user with the help of fresh randomness specific to that user. Such randomization would make the different key components of a user compatible with each other, but not with the parts of a key issued to another user. This is relatively easy to implement in the single-authority setting since there is only one central authority who is responsible to generate secret keys for users.

In a multi-authority, we want to satisfy the simultaneous goals of autonomous key generation and collusion resistance. The requirement of autonomous key

generation means that established techniques for key randomization cannot be applied since there is no one party to compile all the pieces together. Furthermore, in a decentralized MA-ABE system each component may come from a different authority, where such authorities have no coordination and are possibly not even aware of each other. In order to overcome the above challenge, we aim to adapt the high level design rationally of the previous bilinear-map-based decentralized MA-ABE schemes [42,53] to not rely on one key generation call to tie all key components together and instead use the output of a public hash function applied on the user's global identity, GID, as the randomness tying together multiple key components issued by different authorities. However, this means that the randomness responsible for tying together the different key components must be publicly computable, that is, even known to the attacker. Unfortunately, all the CP-ABE schemes realizable under LWE so far fail to satisfy this property.

Importantly, and deviating from previous approaches, we design our CP-ABE scheme carefully so as to have this property. Observe that in our CP-ABE scheme above, the vector $\boldsymbol{t}$ is the one that is used to bind together different key components. A main feature of our CP-ABE scheme is that this vector $\boldsymbol{t}$ is actually part of the output of the key generation procedure. In particular, as we show, the system remains secure *even* when $\boldsymbol{t}$ is public and known to the attacker.

The second challenge in making a CP-ABE scheme compatible for extension to the decentralized multi-authority setting is modularity. Very roughly speaking, the setup and key generation procedures should have the structure such that it should be possible to view their operations as well as their outputs, that is, the master public/secret key and the secret keys of the users as aggregates of individual modules each of which relates to exactly one of the attributes involved. This is important since in a decentralized MA-ABE system, authorities/attributes should be able to join the system at any point of time without requiring any prior coordination with a central authority or a system reset and there is no bound on the number of authorities/attributes that can ever come into existence. Any CP-ABE scheme obtained from an underlying KP-ABE scheme via the universal-circuit-based transformation inherently fails to achieve the above modularity property roughly because in such a system, the master key and the user keys all become associated with the descriptions of circuits rather than the attributes directly. Hence it is not surprising that no prior CP-ABE scheme realizable under LWE achieves the above modularity feature. In contrast, we design our CP-ABE scheme above in such a way that everything is modular and fits into the decentralized multi-authority setting.

As is the design, the proof strategy for our MA-ABE scheme is also somewhat similar to the proof of the CP-ABE scheme. Although, since we are in the multi-authority setting, notation and various technical details become much more involved. For instance, the application of the linear independence property becomes much more delicate. Ignoring notational differences, one additional step we need to make for our proof to go through, is to somehow make the ciphertext components corresponding to corrupted authorities independent of the secret. This is because in our security model, we allow the adversary to generate the

master keys for the corrupted authorities. Hence the simulator cannot hope to program any of the $\boldsymbol{H}_u$ matrices corresponding to the corrupted authorities and thereby cancel the secret present inside those ciphertext components as was possible in the single-authority scheme above.

To solve this, we are inspired by a previous technique of Rouselakis and Waters [53] in the bilinear map world for handling the same problem and we adapt it for our setting. After applying the idea under their transformation we reach a hybrid world which is more similar to the CP-ABE one where we only need to deal with the ciphertext components corresponding to uncorrupted authorities. As an additional contribution, en route to adapting their lemma to our setting, we observe a non-trivial gap in their proof which we resolve (please refer to [25, Section 4.3] for more details).

Lastly, let us explain why the new secret sharing scheme from Sect. 2.1 (see also Theorem 1.4) does not apply here. Since our LSSS from Sect. 2.1 is non-monotone, the share generating matrix has rows for both the positive and negative instances of an attribute. Now, in case of an MA-ABE for non-monotone LSSS, an attacker which corrupts an authority can generate keys for both the positive and negative instances of the attribute controlled by the authority and thus can get hold of both the rows of the LSSS matrix associated with both instances of that attribute. Unfortunately, in our LSSS, the linear independence property only holds when the set of unauthorized rows of an LSSS matrix does not include both the positive and negative instances of a particular attribute simultaneously. (Note that this is not an issue for our CP-ABE scheme since there is only one central authority which remains uncorrupted throughout the system.) We currently do not know of any non-monotone LSSS which achieves the linear independence property even when a set of unauthorized rows include both instances of the same attribute. We therefore settle for an LSSS which only considers attributes in their positive form, that is, monotone LSSS, and still satisfies the linear independence property for unauthorized rows. We use the direct construction of Lewko and Waters [42] which was recently observed by Agrawal et al. [1] to satisfy the linear independence property for unauthorized rows when implemented for the class of DNF formulas.

3 Preliminaries

3.1 Notations

Throughout this paper we will denote the underlying security parameter by λ. A function $\mathsf{negl}\colon \mathbb{N} \to \mathbb{R}$ is *negligible* if it is asymptotically smaller than any inverse-polynomial function, namely, for every constant $c > 0$ there exists an integer N_c such that $\mathsf{negl}(\lambda) \leq \lambda^{-c}$ for all $\lambda > N_c$. We let $[n] = \{1, \ldots, n\}$.

Let PPT stand for probabilistic polynomial-time. For a distribution $\mathcal{X}$, we write $x \leftarrow \mathcal{X}$ to denote that x is sampled at random according to distribution $\mathcal{X}$. For a set X, we write $x \leftarrow X$ to denote that x is sampled according to the uniform distribution over the elements of X. We use bold lower case letters, such

as $\boldsymbol{v}$, to denote vectors and upper-case, such as $\boldsymbol{M}$, for matrices. We assume all vectors, by default, are row vectors. The jth row of a matrix is denoted $\boldsymbol{M}_j$ and analogously for a set of row indices J, we denote $\boldsymbol{M}_J$ for the submatrix of $\boldsymbol{M}$ that consists of the rows $\boldsymbol{M}_j$ for all $j \in J$. For a vector $\boldsymbol{v}$, we let $\|\boldsymbol{v}\|$ denote its ℓ_2 norm and $\|\boldsymbol{v}\|_\infty$ denote its ℓ_∞ norm.

For an integer $q \geq 2$, we let $\mathbb{Z}_q$ denote the ring of integers modulo q. We represent $\mathbb{Z}_q$ as integers in the range $(-q/2, q/2]$.

Indistinguishability: Two sequences of random variables $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ are *computationally indistinguishable* if for any non-uniform PPT algorithm $\mathcal{A}$ there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[\mathcal{A}(1^\lambda, \mathcal{X}_\lambda) = 1] - \Pr[\mathcal{A}(1^\lambda, \mathcal{Y}_\lambda) = 1]| \leq \mathsf{negl}(\lambda)$ for all $\lambda \in \mathbb{N}$.

For two distributions $\mathcal{D}$ and $\mathcal{D}'$ over a discrete domain Ω, the statistical distance between $\mathcal{D}$ and $\mathcal{D}'$ is defined as $\mathsf{SD}(\mathcal{D}, \mathcal{D}') = (1/2)\cdot\sum_{\omega\in\Omega} |\mathcal{D}(\omega) - \mathcal{D}'(\omega)|$. A family of distributions $\mathcal{D} = \{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{D}' = \{\mathcal{D}'_\lambda\}_{\lambda\in\mathbb{N}}$, parameterized by security parameter λ, are said to be *statistically indistinguishable* if there is a negligible function $\mathsf{negl}(\cdot)$ such that $\mathsf{SD}(\mathcal{D}_\lambda, \mathcal{D}'_\lambda) \leq \mathsf{negl}(\lambda)$ for all $\lambda \in \mathbb{N}$.

Smudging: The following lemma says that adding large noise "smudges out" any small values. This lemma was originally proven in [10, Lemma 2.1] and we use a paraphrased version from [35, Lemma 2.1]. Let us first define the notion of a B-bounded distribution.

Definition 3.1 (B-Bounded): For a family of distributions $\mathcal{D} = \{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$ over the integers and a bound $B = B(\lambda) > 0$, we say that $\mathcal{D}$ is *B-bounded* if for every $\lambda \in \mathbb{N}$ it holds that $\Pr_{x\leftarrow\mathcal{D}_\lambda}[|x| \leq B(\lambda)] = 1$.

Lemma 3.1 (Smudging Lemma): *Let $B_1 = B_1(\lambda)$ and $B_2 = B_2(\lambda)$ be positive and let $\mathcal{D} = \{\mathcal{D}_\lambda\}_\lambda$ be a B_1-bounded distribution family. Let $\mathcal{U} = \{\mathcal{U}_\lambda\}_\lambda$ be the uniform distribution over $[-B_2(\lambda), B_2(\lambda)]$. The family of distributions $\mathcal{D}+\mathcal{U}$ and $\mathcal{U}$ are statistically indistinguishable if there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ it holds that $B_1(\lambda)/B_2(\lambda) \leq \mathsf{negl}(\lambda)$.*

Leftover Hash Lemma: We recall the well known leftover hash lemma, stated in a convenient form for our needs (e.g., [2,52]).

Lemma 3.2 (Leftover Hash Lemma): *Let $n\colon \mathbb{N} \to \mathbb{N}$, $q\colon \mathbb{N} \to \mathbb{N}$, $m > (n+1)\log q + \omega(\log n)$, and $k = k(n)$ be some polynomial. Then, the following two distributions are statistically indistinguishable:*

$$\mathcal{D}_1 \equiv \left\{ (\boldsymbol{A}, \boldsymbol{AR}) \mid \boldsymbol{A} \leftarrow \mathbb{Z}_q^{n\times m}, \boldsymbol{R} \leftarrow \{-1,1\}^{m\times k} \right\},$$
$$\mathcal{D}_2 \equiv \left\{ (\boldsymbol{A}, \boldsymbol{S}) \mid \boldsymbol{A} \leftarrow \mathbb{Z}_q^{n\times m}, \boldsymbol{S} \leftarrow \mathbb{Z}_q^{n\times k} \right\}.$$

3.2 Lattice and LWE Preliminaries

Here, we provide necessary background on lattices, the LWE assumption, and various useful tools that we use.

Lattices: An m-dimensional lattice $\mathcal{L}$ is a discrete additive subgroup of $\mathbb{R}^m$. Given positive integers n, m, q and a matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$, we let $\lambda_q^{\perp}(\boldsymbol{A})$ denote the lattice $\{\boldsymbol{x} \in \mathbb{Z}^m \mid \boldsymbol{A}\boldsymbol{x}^\top = \boldsymbol{0}^\top \bmod q\}$. For $\boldsymbol{u} \in \mathbb{Z}_q^n$, we let $\lambda_q^{\boldsymbol{u}}(\boldsymbol{A})$ denote the coset $\{\boldsymbol{x} \in \mathbb{Z}^m \mid \boldsymbol{A}\boldsymbol{x}^\top = \boldsymbol{u}^\top \bmod q\}$.

Discrete Gaussians: Let σ be any positive real number. The Gaussian distribution $\mathcal{D}_\sigma$ with parameter σ is defined by the probability distribution function $\rho_\sigma(\boldsymbol{x}) = \exp(-\pi\|x\|^2/\sigma^2)$. For any discrete set $\mathcal{L} \subseteq \mathbb{R}^m$, define $\rho_\sigma(\mathcal{L}) = \sum_{\boldsymbol{x} \in \mathcal{L}} \rho_\sigma(\boldsymbol{x})$. The discrete Gaussian distribution $\mathcal{D}_{\mathcal{L},\sigma}$ over $\mathcal{L}$ with parameter σ is defined by the probability distribution function $\rho_{\mathcal{L},\sigma}(\boldsymbol{x}) = \rho_\sigma(\boldsymbol{x})/\rho_\sigma(\mathcal{L})$.

The following lemma (e.g., [47, Lemma 4.4]) shows that if the parameter σ of a discrete Gaussian distribution is small, then any vector drawn from this distribution will be short (with high probability).

Lemma 3.3: *Let m, n, q be positive integers with $m > n$, $q > 2$. Let $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$ be a matrix of dimensions $n \times m$, $\sigma = \tilde{\Omega}(n)$, and $\mathcal{L} = \lambda_q^{\perp}(\boldsymbol{A})$. Then, there is a negligible function $\mathsf{negl}(\cdot)$ such that*

$$\Pr_{\boldsymbol{x} \leftarrow \mathcal{D}_{\mathcal{L},\sigma}} \left[\|\boldsymbol{x}\| > \sqrt{m}\sigma\right] \leq \mathsf{negl}(n),$$

where $\|\boldsymbol{x}\|$ denotes the ℓ_2 norm of $\boldsymbol{x}$.

Truncated Discrete Gaussians: The truncated discrete Gaussian distribution over $\mathbb{Z}^m$ with parameter σ, denoted by $\widetilde{\mathcal{D}}_{\mathbb{Z}^m,\sigma}$, is the same as the discrete Gaussian distribution $\mathcal{D}_{\mathbb{Z}^m,\sigma}$ except that it outputs 0 whenever the ℓ_∞ norm exceeds $\sqrt{m}\sigma$. Note that, by definition, $\widetilde{\mathcal{D}}_{\mathbb{Z}^m,\sigma}$ is $\sqrt{m}\sigma$-bounded. Also, by Lemma 3.3 we get that $\widetilde{\mathcal{D}}_{\mathbb{Z}^m,\sigma}$ and $\mathcal{D}_{\mathbb{Z}^m,\sigma}$ are statistically indistinguishable.

3.2.1 Lattice Trapdoors

Lattices with trapdoors are lattices that are indistinguishable from randomly chosen lattices, but have certain "trapdoors" that allow efficient solutions to hard lattice problems. A trapdoor lattice sampler [9,29,45], denoted $\mathsf{LT} = (\mathsf{TrapGen}, \mathsf{SamplePre})$, consists of two algorithms with the following syntax and properties:

- $\mathsf{TrapGen}(1^n, 1^m, q) \mapsto (\boldsymbol{A}, T_{\boldsymbol{A}})$: The lattice generation algorithm is a randomized algorithm that takes as input the matrix dimensions n, m, modulus q, and outputs a matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$ together with a trapdoor $T_{\boldsymbol{A}}$.
- $\mathsf{SamplePre}(\boldsymbol{A}, T_{\boldsymbol{A}}, \sigma, \boldsymbol{u}) \mapsto \boldsymbol{s}$: The presampling algorithm takes as input a matrix $\boldsymbol{A}$, trapdoor $T_{\boldsymbol{A}}$, a vector $\boldsymbol{u} \in \mathbb{Z}_q^n$, and a parameter $\sigma \in \mathbb{R}$ (which determines the length of the output vectors). It outputs a vector $\boldsymbol{s} \in \mathbb{Z}_q^m$ such that $\boldsymbol{A} \cdot \boldsymbol{s}^\top = \boldsymbol{u}^\top$ and $\|\boldsymbol{s}\| \leq \sqrt{m} \cdot \sigma$.

Well-sampledness: Following Goyal et al. [35], we further require that the aforementioned sampling procedures output well-sampled elements. That is, the

1. The adversary $\mathcal{A}$ receives input 1^λ and sends $1^n, 1^m, 1^z$ such that $m > n \log q(\lambda) + \lambda$ to the challenger.
2. Upon receipt, the challenger first selects a random bit $b \leftarrow \{0,1\}$. Next, it samples $\left\{(\boldsymbol{A}_{i,0}, T_{\boldsymbol{A}_{i,0}})\right\}_{i\in[z]} \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ and $\{\boldsymbol{A}_{i,1}\}_{i\in[z]} \leftarrow \mathbb{Z}_q^{n\times m}$. It sends $\{\boldsymbol{A}_{i,b}\}_{i\in[z]}$ to $\mathcal{A}$.
3. Finally, $\mathcal{A}$ outputs its guess $b' \in \{0,1\}$. The experiment outputs 1 if and only if $b = b'$.

Fig. 1. $\mathsf{Exp}^{\mathsf{matrix},q}_{\mathsf{LT},\mathcal{A}}$

matrix outputted by $\mathsf{TrapGen}$ looks like a uniformly random matrix, and the preimage outputted by $\mathsf{SamplePre}$ with a uniformly random vector/matrix is indistinguishable from a vector/matrix with entries drawn from an appropriate Gaussian distribution. These two properties are summarized next.

Definition 3.2 (Well-Sampledness of Matrix): Fix any function $q\colon \mathbb{N} \to \mathbb{N}$. The procedure $\mathsf{TrapGen}$ is said to satisfy the *q-well-sampledness of matrix* property if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\mathsf{Adv}^{\mathsf{matrix},q}_{\mathsf{LT},\mathcal{A}}(\lambda) \triangleq \left|\mathsf{Pr}\left[\mathsf{Exp}^{\mathsf{matrix},q}_{\mathsf{LT},\mathcal{A}}(\lambda) = 1\right] - 1/2\right| \leq \mathsf{negl}(\lambda),$$

where $\mathsf{Exp}^{\mathsf{matrix},q}_{\mathsf{LT},\mathcal{A}}(\lambda)$ is defined in Fig. 1.

Definition 3.3 (Well-Sampledness of Preimage): Fix any function $q\colon \mathbb{N} \to \mathbb{N}$ and $\sigma\colon \mathbb{N} \to \mathbb{N}$. The procedure $\mathsf{SamplePre}$ is said to satisfy the *(q,σ)-well-sampledness* property if for any stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\mathsf{Adv}^{\mathsf{preimage},q,\sigma}_{\mathsf{LT},\mathcal{A}}(\lambda) \triangleq \left|\mathsf{Pr}\left[\mathsf{Exp}^{\mathsf{preimage},q,\sigma}_{\mathsf{LT},\mathcal{A}}(\lambda) = 1\right] - 1/2\right| \leq \mathsf{negl}(\lambda),$$

where $\mathsf{Exp}^{\mathsf{preimage},q,\sigma}_{\mathsf{LT},\mathcal{A}}$ is defined in Fig. 2.

Both the above properties are satisfied by the gadget-based trapdoor lattice sampler presented in [45].

Enhanced trapdoor sampling: Let $q\colon \mathbb{N} \to \mathbb{N}$, $\sigma\colon \mathbb{N} \to \mathbb{R}^+$ be functions and $\mathsf{LT} = (\mathsf{TrapGen}, \mathsf{SamplePre})$ be a trapdoor lattice sampler satisfying the q-well-sampledness of matrix and (q,σ)-well-sampledness of preimage properties. We describe enhanced trapdoor lattice sampling algorithms $\mathsf{EnLT} = (\mathsf{EnTrapGen}, \mathsf{EnSamplePre})$ due to Goyal et al. [35] (which are, in turn, reminiscent of the trapdoor extension algorithms of [3,20]).

1. The adversary $\mathcal{A}$ receives input 1^λ and sends $1^n, 1^m, 1^z$ such that $\sigma(\lambda) > \sqrt{n \cdot \log q(\lambda) \cdot \log m} + \lambda$ and $m > n \cdot \log q(\lambda) + \lambda$ to the challenger.
2. Upon receipt, the challenger first selects a random bit $b \leftarrow \{0,1\}$. Next, it samples $\{(\boldsymbol{A}_i, T_{\boldsymbol{A}_i})\}_{i\in[z]} \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ and sends $\{\boldsymbol{A}_i\}_{i\in[z]}$ to $\mathcal{A}$.
3. Then, $\mathcal{A}$ makes a $\mathsf{poly}(\lambda)$ number of pre-image queries of the form $i \in [z]$ to the challenger and the challenger responds as follows:
   (a) It samples $\boldsymbol{w} \leftarrow \mathbb{Z}_q^n$, $\boldsymbol{u}_0 \leftarrow \mathsf{SamplePre}(\boldsymbol{A}_i, T_{\boldsymbol{A}_i}, \sigma, \boldsymbol{w})$, and $\boldsymbol{u}_1 \leftarrow \mathcal{D}^m_{\mathbb{Z},\sigma}$. It sends $\boldsymbol{u}_b$ to $\mathcal{A}$.
4. Finally, $\mathcal{A}$ outputs its guess $b' \in \{0,1\}$. The experiment outputs 1 if and only if $b = b'$.

Fig. 2. $\mathsf{Exp}^{\mathsf{preimage},q,\sigma}_{\mathsf{LT},\mathcal{A}}$

- $\mathsf{EnTrapGen}(1^n, 1^m, q) \mapsto (\boldsymbol{A}, T_{\boldsymbol{A}})$: The trapdoor generation algorithm generates two matrices $\boldsymbol{A}_1 \in \mathbb{Z}_q^{n\times\lceil m/2\rceil}$ and $\boldsymbol{A}_2 \in \mathbb{Z}_q^{n\times\lfloor m/2\rfloor}$ as $(\boldsymbol{A}_1, T_{\boldsymbol{A}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^{\lceil m/2\rceil}, q)$, $(\boldsymbol{A}_2, T_{\boldsymbol{A}_2}) \leftarrow \mathsf{TrapGen}(1^n, 1^{\lfloor m/2\rfloor}, q)$. It appends both matrices column-wise to obtain a larger matrix $\boldsymbol{A}$ as $\boldsymbol{A} = (\boldsymbol{A}_1|\boldsymbol{A}_2)$ and sets the associated trapdoor $T_{\boldsymbol{A}}$ to be the combined trapdoor information $T_{\boldsymbol{A}} = (T_{\boldsymbol{A}_1}, T_{\boldsymbol{A}_2})$.
- $\mathsf{EnSamplePre}(\boldsymbol{A}, T_{\boldsymbol{A}}, \sigma, \boldsymbol{Z}) \mapsto \boldsymbol{S}$: The pre-image sampling algorithm takes as input a matrix $\boldsymbol{A} = (\boldsymbol{A}_1|\boldsymbol{A}_2)$ with trapdoor $T_{\boldsymbol{A}} = (T_{\boldsymbol{A}_1}, T_{\boldsymbol{A}_2})$, a parameter $\sigma = \sigma(\lambda)$, and a matrix $\boldsymbol{Z} \in \mathbb{Z}_q^{n\times k}$. It chooses a uniformly random matrix $\boldsymbol{W} \leftarrow \mathbb{Z}_q^{n\times k}$ and sets $\boldsymbol{Y} = \boldsymbol{Z} - \boldsymbol{W}$. Next, it computes matrices $\boldsymbol{S}_1, \boldsymbol{S}_2 \in \mathbb{Z}^{\lceil m/2\rceil\times k}$ as $\boldsymbol{S}_1 \leftarrow \mathsf{SamplePre}(\boldsymbol{A}_1, T_{\boldsymbol{A}_1}, \sigma, \boldsymbol{W})$ and $\boldsymbol{S}_2 \leftarrow \mathsf{SamplePre}(\boldsymbol{A}_2, T_{\boldsymbol{A}_2}, \sigma, \boldsymbol{Y})$. It computes the final output matrix $\boldsymbol{S} \in \mathbb{Z}^{m\times k}$ by column-wise appending matrices $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ as $\boldsymbol{S} = (\boldsymbol{S}_1|\boldsymbol{S}_2)$.

The well-sampledness properties (Definition 3.2 and Definition 3.3) of EnLT are inherited from the same properties of the underlying LT [35, Section 7.3].

We show that the enhanced trapdoor sampling procedures EnLT satisfy another property (which as far as we know has not been used or formalized before). We refer this property as "leftover hash lemma with trapdoors". This property is crucial in the security proofs of our constructions. Recall that in the original leftover hash lemma (Lemma 3.2 above) the matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$ appearing in the two indistinguishable distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ is sampled uniformly at random. The "leftover hash lemma with trapdoors" property of EnLT basically states that the leftover hash lemma holds even when the matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$ is generated by the $\mathsf{EnTrapGen}$ algorithm and is not uniformly random. (See the full version [25] for the formal description and proof of the lemma.)

3.2.2 Learning with Errors

Assumption 1 (Learning With Errors (LWE) [52]): For a security parameter $\lambda \in \mathbb{N}$, let $n \colon \mathbb{N} \to \mathbb{N}$, $q \colon \mathbb{N} \to \mathbb{N}$, and $\sigma \colon \mathbb{N} \to \mathbb{R}^+$ be functions of λ. The Learning with Errors (LWE) assumption $\mathsf{LWE}_{n,q,\sigma}$, parametrized by $n = n(\lambda), q = q(\lambda), \sigma = \sigma(\lambda)$, states that for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$,

$$\begin{aligned}\mathsf{Adv}_{\mathcal{A}}^{\mathsf{LWE}_{n,q,\sigma}}(\lambda) &\triangleq \left|\mathsf{Pr}\left[1 \leftarrow \mathcal{A}^{\mathcal{O}_1^{\boldsymbol{s}}(\cdot)}(1^\lambda) \mid \boldsymbol{s} \leftarrow \mathbb{Z}_q^n\right] - \mathsf{Pr}\left[1 \leftarrow \mathcal{A}^{\mathcal{O}_2(\cdot)}(1^\lambda)\right]\right| \\ &\leq \mathsf{negl}(\lambda),\end{aligned}$$

where the oracles $\mathcal{O}_1^{\boldsymbol{s}}(\cdot)$ and $\mathcal{O}_2(\cdot)$ are defined as follows: $\mathcal{O}_1^{\boldsymbol{s}}(\cdot)$ has $\boldsymbol{s} \in \mathbb{Z}_q^n$ hardwired, and on each query it chooses $\boldsymbol{a} \leftarrow \mathbb{Z}_q^n$, $e \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}$ and outputs $(\boldsymbol{a}, \boldsymbol{s}\boldsymbol{a}^\top + e \bmod q)$, and $\mathcal{O}_2(\cdot)$ on each query chooses $\boldsymbol{a} \leftarrow \mathbb{Z}_q^n$, $u \leftarrow \mathbb{Z}_q$ and outputs $(\boldsymbol{a}, u)$.

Regev [52] showed that if there exists a PPT adversary that can break the LWE assumption, then there exists a PPT quantum algorithm that can solve some hard lattice problems in the worst case. Given the current state of the art of lattice problems [17,29,46,47,51,52], the LWE assumption is believed to be true for any polynomial $n(\cdot)$ and any functions $q(\cdot)$, $\sigma(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $n = n(\lambda), q = q(\lambda), \sigma = \sigma(\lambda)$ satisfy the following constraints:

$$2\sqrt{n} < \sigma < q < 2^n, \quad n \cdot q/\sigma < 2^{n^\epsilon}, \quad \text{and } 0 < \epsilon < 1/2$$

4 Linear Secret Sharing Schemes with Linear Independence

In this section, we first provide the necessary definitions and properties of linear secret sharing schemes. Then, we present a new linear secret sharing scheme for all non-monotone access structures realizable by NC^1 circuits. This new secret sharing scheme has some interesting properties which we crucially utilize while designing our CP-ABE scheme for all NC^1 circuits under the LWE assumption.

4.1 Background on Linear Secret Sharing Schemes

A secret sharing scheme consists of a dealer who holds a secret and a set of n parties. Informally, the dealer "splits" the secret into "shares" and distributes them among the parties. Subsets of parties which are "authorized" should be able to jointly recover the secret while others should not. The description of the set of authorized sets is called the *access structure*.

Definition 4.1 (Access Structures): An access structure on n parties associated with numbers in $[n]$ is a set $\mathbb{A} \subseteq 2^{[n]} \setminus \emptyset$ of non-empty subsets of parties. The sets in $\mathbb{A}$ are called the *authorized* sets and the sets not in $\mathbb{A}$ are called the *unauthorized* sets. An access structure is called *monotone* if $\forall B, C \in 2^{[n]}$ if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$.

A secret sharing scheme for a monotone access structure $\mathbb{A}$ is a randomized algorithm that on input a secret z outputs n shares $\mathsf{sh}_1, \ldots, \mathsf{sh}_n$ such that for any $A \in \mathbb{A}$ the shares $\{\mathsf{sh}_i\}_{i \in A}$ determine z and other sets are independent of z (as random variables).

Non-monotone secret sharing: A natural generalization of the above notion that captures all access structures (rather than only monotone ones) is called *non-monotone* secret sharing. Concretely, a non-monotone secret sharing scheme for an access structure $\mathbb{A}$ is a randomized algorithm that on input a secret z outputs $2n$ shares viewed as n pairs $(\mathsf{sh}_{1,0}, \mathsf{sh}_{1,1}), \ldots, (\mathsf{sh}_{n,0}, \mathsf{sh}_{n,1})$ such that for any $A \in \mathbb{A}$ the shares $\{\mathsf{sh}_{i,1}\}_{i \in A} \cup \{\mathsf{sh}_{i,0}\}_{i \notin A}$ determine z and other sets are independent of z.

We will be interested in a subset of all (non-monotone) secret sharing schemes where the reconstruction procedure is a linear function of the shares [38]. These are known as *linear* (*non-monotone*) *secret sharing schemes*.

Definition 4.2 (Linear (non-monotone) secret sharing schemes): Let $q \in \mathbb{N}$ be a prime power and $[n]$ be a set of parties. A non-monotone secret-sharing scheme Π with domain of secrets $\mathbb{Z}_q$ realizing access structure $\mathbb{A}$ on parties $[n]$ is linear over $\mathbb{Z}_q$ if

1. Each share $\mathsf{sh}_{i,b}$ for $i \in [n]$ and $b \in \{0,1\}$ of a secret $z \in \mathbb{Z}_q$ forms a vector with entries in $\mathbb{Z}_q$.
2. There exists a matrix $\boldsymbol{M} \in \mathbb{Z}_q^{\ell \times d}$, called the share-generating matrix, and a function $\rho\colon [\ell] \to [2n]$, that labels the rows of $\boldsymbol{M}$ with a party index from $[n]$ or its corresponding negation, represented as another party index from $\{n+1, \ldots, 2n\}$, which satisfy the following: During the generation of the shares, we consider the vector $\boldsymbol{v} = (z, r_2, ..., r_d) \in \mathbb{Z}_q^d$. Then the vector of ℓ shares of the secret z according to Π is equal to $\mathbf{sh} = \boldsymbol{M} \cdot \boldsymbol{v}^\top \in \mathbb{Z}_q^{\ell \times 1}$. For $i \in [n]$ and $b \in \{0,1\}$, the share $\mathbf{sh}_{i,b}$ consists of all $\mathbf{sh}_j$ values for which $\rho(j) = n \cdot (1-b) + i$ (so the first n shares correspond to the "1 shares" and the last n shares correspond to the "0 shares").
   We will be referring to the pair $(\boldsymbol{M}, \rho)$ as the LSSS *policy* of the access structure $\mathbb{A}$.

It is well known that the above method of sharing a secret satisfies the desired correctness and security of a non-monotone secret sharing scheme as defined above (e.g., [38]). For an LSSS policy $(\boldsymbol{M}, \rho)$, where $\boldsymbol{M} \in \mathbb{Z}_q^{\ell \times d}$ and $\rho\colon [\ell] \to [2n]$, and a set of parties $S \subseteq [n]$, let $\widehat{S} = S \cup \{i \in \{n+1, \ldots, 2n\} \mid i - n \notin S\} \subset [2n]$. We denote $\boldsymbol{M}_{\widehat{S}}$ the submatrix of $\boldsymbol{M}$ that consists of all the rows of $\boldsymbol{M}$ that "belong" to $\widehat{S}$ according to ρ (i.e., rows j for which $\rho(j) \in \widehat{S}$). *Correctness* means that if $S \subseteq [n]$ is authorized, the vector $(1, \overbrace{0, \ldots, 0}^{d-1}) \in \mathbb{Z}_q^d$ is in the span of the rows of $\boldsymbol{M}_{\widehat{S}}$. *Security* means that if $S \subseteq [n]$ is unauthorized, the vector $(1, 0, \ldots, 0)$ is *not* in the span of the rows of $\boldsymbol{M}_{\widehat{S}}$. Also, in the unauthorized case, there exists a vector $\boldsymbol{d} \in \mathbb{Z}_q^d$, such that its first component $\boldsymbol{d}_1 = 1$ and $\boldsymbol{M}_{\widehat{S}} \boldsymbol{d}^\top = \mathbf{0}$, where $\mathbf{0}$ is the all 0 vector.

{0, 1}-LSSS: A special subset of all linear secret sharing schemes are ones where the reconstruction coefficients are always binary [14, Definition 4.13]. We call such LSSS a $\{0,1\}$-LSSS. This property of LSSS secret sharing schemes was recently formally defined by [14]. They observed that a well-known construction by Lewko and Waters [42] actually results with an LSSS with this property for all access structures in NC^1.

On sharing vectors: The above sharing and reconstruction methods directly extend to sharing a vector $\boldsymbol{z} \in \mathbb{Z}_q^m$ of dimension $m \in \mathbb{N}$ rather than just scalars.

4.2 Our Non-monotone LSSS for NC^1

We introduce a new non-monotone linear secret sharing scheme for all access structures that can be described by NC^1 circuits. The new scheme has some useful properties for us which we summarize next:

- The entries in the corresponding policy matrix are small, i.e., coming from $\{-1, 0, 1\}$.
- Reconstruction of the secret can be done by small coefficients, i.e., coming from $\{0, 1\}$.
- The rows of the corresponding policy matrix that correspond to an unauthorized set are *linearly independent*.

Remark 4.1: The well-known construction of Lewko and Waters [42] actually results with an LSSS with these properties for all access structures described by DNF formulas. This was recently observed by [1]. As opposed to our construction, this construction is a monotone LSSS, not a non-monotone one.

The construction: We are given an access structure $\mathbb{A}$ described by an NC^1 circuit. This circuit can be described by a Boolean formula of logarithmic depth that consists of (fan-in 2) AND, OR, and (fan-in 1) NOT gates. We further push the NOT gates to the leaves using De Morgan laws, and from now on we assume that internal nodes only constitute of OR and AND gates and leaves are labeled either by variables or their negations. In other words, we assume that we are given a monotone Boolean formula consisting only of AND and OR gates. We would like to highlight that even if we are starting off with a monotone Boolean formula, the LSSS secret sharing scheme we are going to construct would be a non-monotone one. More precisely, the algorithm associates with each input variable x_i of the monotone Boolean formula two vector shares $\mathbf{sh}_{i,0}$ and $\mathbf{sh}_{i,1}$. This is done in a recursive fashion starting from the root by associating with each internal wire w two labels $\boldsymbol{w}_1$ and $\boldsymbol{w}_0$ (and the labels of the leaves correspond to the shares). The labels of the root w are $\boldsymbol{w}_1 = (1, 0, \ldots, 0)$ and $\boldsymbol{w}_0 = (0, 1, 0, \ldots, 0)$, both of which are of dimension $\tilde{k} \triangleq k + 2$, where k is the number of gates in the formula. We maintain a global counter variable c which is initialized to 2 and is increased by one after labeling each gate. We shall traverse the tree from top (root) to bottom (leaves) and within a layer from left to right. Consider a gate whose output wire w labels are $\boldsymbol{w}_1$, $\boldsymbol{w}_0$ and denote its children wires, u and v,

with corresponding labels (to be assigned) $\boldsymbol{u}_1, \boldsymbol{u}_0$ and $\boldsymbol{v}_1, \boldsymbol{v}_0$, respectively. The assignment is done as follows, depending on the type of the gate connecting u and v to w:

AND gate: $\boldsymbol{u}_1 = 0^c \| 1 \| 0^{\tilde{k}-c-1}$, $\boldsymbol{u}_0 = \boldsymbol{w}_0$, $\boldsymbol{v}_1 = \boldsymbol{w}_1 - \boldsymbol{u}_1$, $\boldsymbol{v}_0 = \boldsymbol{w}_0 - \boldsymbol{u}_1$.
OR gate: $\boldsymbol{u}_1 = \boldsymbol{w}_1$, $\boldsymbol{u}_0 = 0^c \| 1 \| 0^{\tilde{k}-c-1}$, $\boldsymbol{v}_1 = \boldsymbol{w}_1 - \boldsymbol{u}_0$, $\boldsymbol{v}_0 = \boldsymbol{w}_0 - \boldsymbol{u}_0$.

An example: Consider the monotone Boolean formula $(A \wedge B) \vee (C \wedge D)$. The 1-label of the root is $(1,0,0,0,0)$ and the 0-label is $(0,1,0,0,0)$. The 1-label of the left child of the OR gate is $(1,0,0,0,0)$ and the 0-label is $(0,0,1,0,0)$. The 1-label of the right child of the OR gate is $(1,0,-1,0,0)$ and the 0-label is $(0,1,-1,0,0)$. Therefore, the resulting policy is

$$\boldsymbol{M} = \begin{array}{c} A_1 \\ A_0 \\ B_1 \\ B_0 \\ C_1 \\ C_0 \\ D_1 \\ D_0 \end{array} \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 & -1 \\ 0 & 1 & -1 & 0 & -1 \end{pmatrix}$$

The following lemma follows by induction on the number of gates in the formula. Recall that for $S \subseteq [n]$, we let $\widehat{S} = S \cup \{i \in \{n+1, \ldots, 2n\} \mid i - n \notin S\} \subset [2n]$ and let $\boldsymbol{M}_{\widehat{S}}$ be the submatrix that consists of all the rows of $\boldsymbol{M}$ that "belong" to $\widehat{S}$ according to ρ.

Lemma 4.1: *For any access structure $\mathbb{A}$ which is described by a Boolean formula, the above process for generating the matrix $\boldsymbol{M}$ results with*

1. *A non-monotone $\{0,1\}$-LSSS for $\mathbb{A}$, namely*
   (a) *For any authorized set of parties $S \subseteq [n]$, there is a linear combination of the rows of $\boldsymbol{M}_{\widehat{S}}$ that results with $(1,0,\ldots,0) \in \mathbb{Z}_q^d$. Moreover, the coefficients in this linear combination are from $\{0,1\}$.*
   (b) *For any unauthorized set of parties $S \subseteq [n]$, no linear combination of the rows of $\boldsymbol{M}_{\widehat{S}}$ results in $(1,0,\ldots,0) \in \mathbb{Z}_q^d$. Also, there exists a vector $\boldsymbol{d} \in \mathbb{Z}_q^d$, such that its first component $\boldsymbol{d}_1 = 1$ and $\boldsymbol{M}_{\widehat{S}} \boldsymbol{d}^\top = \boldsymbol{0}$, where $\boldsymbol{0}$ is the all 0 vector.*
2. *For any unauthorized set of parties $S \subseteq [n]$, all of the rows of $\boldsymbol{M}_{\widehat{S}}$ are linearly independent.*

The proof of Lemma 4.1 can be found in the full version [25, Section 4.2].

5 Our Ciphertext-Policy ABE Scheme

In this section, we present our ciphertext-policy ABE (CP-ABE) scheme supporting access structures represented by NC^1 circuits. The scheme is associated

with a fixed attribute universe $\mathbb{U}$ and we will use the transformation described in Sect. 4.2 to represent the access structures as non-monotone LSSS. More precisely, we only design a CP-ABE scheme for LSSS access policies $(\boldsymbol{M}, \rho)$ with properties stipulated in Lemma 4.1, that is, we construct a CP-ABE scheme for LSSS access policies $(\boldsymbol{M}, \rho)$ such that the entries of $\boldsymbol{M}$ come from $\{-1, 0, 1\}$ as well as reconstruction only involves coefficients coming from $\{0, 1\}$, and prove the scheme to be selectively secure under linear independence restriction (see [25, Definition 3.5] in the full version for the formal description of the security model). It then follows directly from Lemma 4.1, that our CP-ABE scheme actually achieves the standard notion of selective security (see [25, Definition 3.4] in the full version for the formal description of the security model) when implemented for the class of all access structures represented by NC^1 circuits. Further, we will assume that all LSSS access policies $(\boldsymbol{M}, \rho)$ used in our scheme correspond to matrices $\boldsymbol{M}$ with at most $s_{\max}$ columns and an injective row-labeling function ρ, i.e., an attribute is associated with at most one row of $\boldsymbol{M}$.[2] Since our Boolean formula to LSSS transformation from Sect. 4.2 generates a new column in the resulting LSSS matrix for each gate in the underlying Boolean formula, the bound $s_{\max}$ on the number of columns in our CP-ABE construction naturally translates to a bound on the circuit size of the supported NC^1 access policies at implementation. Also, in our scheme description below, we assume for simplicity of presentation that both the encryption and the decryption algorithms receive an access policy directly in its LSSS representation. However, we note that in the actual implementation, the encryption and decryption algorithms should instead take in the circuit representation of the access policy and deterministically compute its LSSS representation using our transformation algorithm from Sect. 4.2. This is because, without the circuit description of an access policy, the decryption algorithm may not be able to efficiently determine the $\{0, 1\}$ reconstruction coefficients needed for a successful decryption.

First, we provide the parameter constraints required by our correctness and security proof. Fix any $0 < \epsilon < 1/2$. For any $B \in \mathbb{N}$, let $\mathcal{U}_B$ denote the uniform distribution on $\mathbb{Z} \cap [-B, B]$, i.e., integers between $\pm B$. The Setup algorithm chooses parameters n, m, σ, q and noise distributions $\chi_{\mathsf{lwe}}, \chi_1, \chi_2, \chi_{\mathsf{big}}$, satisfying the following constraints:

- $n = \mathsf{poly}(\lambda)$, $\sigma < q$, $n \cdot q/\sigma < 2^{n^\epsilon}$, $\chi_{\mathsf{lwe}} = \widetilde{\mathcal{D}}_{\mathbb{Z},\sigma}$ (for LWE security)
- $m > 2s_{\max} n \log q + \omega \log n + 2\lambda$ (for enhanced trapdoor sampling and LHL)
- $\sigma > \sqrt{s_{\max} n \log q \log m} + \lambda$ (for enhanced trapdoor sampling)
- $\chi_1 = \widetilde{\mathcal{D}}_{\mathbb{Z}^{m-1},\sigma}$, $\chi_2 = \widetilde{\mathcal{D}}_{\mathbb{Z}^m,\sigma}$ (for enhanced trapdoor sampling)
- $\chi_{\mathsf{big}} = \mathcal{U}_{\hat{B}}$, where $\hat{B} > (m^{3/2}\sigma + 1)2^\lambda$ (for smudging/security)
- $|\mathbb{U}| \cdot 3m^{3/2}\sigma\hat{B} < q/4$ (for correctness)

Now, we describe our CP-ABE construction.

[2] Note that following the simple encoding technique devised in [42,58], we can alleviate the injective restriction on the row labeling functions to allow an attribute to appear an a priori bounded number of times within the LSSS access policies.

Setup$(1^\lambda, s_{\max}, \mathbb{U})$**:** The setup algorithm takes in the security parameter λ encoded in unary, the maximum width $s_{\max} = s_{\max}(\lambda)$ of an LSSS matrix supported by the scheme, and the attribute universe $\mathbb{U}$ associated with the system. It first chooses an LWE modulus q, dimensions n, m, and also distributions $\chi_{\mathsf{lwe}}, \chi_1, \chi_2, \chi_{\mathsf{big}}$ as described above. Next, it chooses a vector $\boldsymbol{y} \leftarrow \mathbb{Z}_q^n$ and a sequence of matrices $\{\boldsymbol{H}_u\}_{u\in\mathbb{U}} \leftarrow \mathbb{Z}_q^{n\times m}$. Then, it samples pairs of matrices with trapdoors $\{(\boldsymbol{A}_u, T_{\boldsymbol{A}_u})\}_{u\in\mathbb{U}} \leftarrow \mathsf{EnTrapGen}(1^n, 1^m, q)$. Finally, it outputs

$$\mathsf{PK} = \left(n, m, q, \chi_{\mathsf{lwe}}, \chi_1, \chi_2, \chi_{\mathsf{big}}, \boldsymbol{y}, \{\boldsymbol{A}_u\}_{u\in\mathbb{U}}, \{\boldsymbol{H}_u\}_{u\in\mathbb{U}}\right), \qquad \mathsf{MSK} = \{T_{\boldsymbol{A}_u}\}_{u\in\mathbb{U}}.$$

KeyGen$(\mathsf{MSK}, \boldsymbol{U})$**:** The key generation algorithm takes as input the master secret key MSK, and a set of attributes $U \subseteq \mathbb{U}$. It samples a vector $\hat{\boldsymbol{t}} \leftarrow \chi_1$ and sets the vector $\boldsymbol{t} = (1, \hat{\boldsymbol{t}}) \in \mathbb{Z}^m$. For each $u \in U$, it samples vectors $\hat{\boldsymbol{k}}_u \leftarrow \chi_{\mathsf{big}}^m$ and $\tilde{\boldsymbol{k}}_u \leftarrow \mathsf{EnSamplePre}(\boldsymbol{A}_u, T_{\boldsymbol{A}_u}, \sigma, \boldsymbol{t}\boldsymbol{H}_u^\top - \hat{\boldsymbol{k}}_u\boldsymbol{A}_u^\top)$, and sets $\boldsymbol{k}_u = \hat{\boldsymbol{k}}_u + \tilde{\boldsymbol{k}}_u$. Finally, it outputs

$$\mathsf{SK} = \left(\{\boldsymbol{k}_u\}_{u\in U}, \boldsymbol{t}\right).$$

Enc$(\mathsf{PK}, \mathsf{msg}, (\boldsymbol{M}, \boldsymbol{\rho}))$**:** The encryption algorithm takes as input the public parameters PK, a message $\mathsf{msg} \in \{0,1\}$ to encrypt, and an LSSS access policy $(\boldsymbol{M}, \rho)$ generated by the transformation from Sect. 4.2, where $\boldsymbol{M} = (M_{i,j})_{\ell\times s_{\max}} \in \{-1, 0, 1\}^{\ell\times s_{\max}} \subset \mathbb{Z}_q^{\ell\times s_{\max}}$ (Lemma 4.1) and $\rho\colon [\ell] \to \mathbb{U}$. The function ρ associates rows of $\boldsymbol{M}$ to attributes in $\mathbb{U}$. We assume that ρ is an injective function. The procedure samples vectors $\boldsymbol{s} \leftarrow \mathbb{Z}_q^n$ and $\{\boldsymbol{v}_j\}_{j\in\{2,\ldots,s_{\max}\}} \leftarrow \mathbb{Z}_q^m$. It additionally samples vectors $\{\boldsymbol{e}_i\}_{i\in[\ell]} \leftarrow \chi_{\mathsf{lwe}}^m$ and $\{\hat{\boldsymbol{e}}_i\}_{i\in[\ell]} \leftarrow \chi_{\mathsf{big}}^m$. For each $i \in [\ell]$, it computes vectors $\boldsymbol{c}_i, \hat{\boldsymbol{c}}_i \in \mathbb{Z}_q^m$ as follows:

$$\begin{aligned}
\boldsymbol{c}_i &= \boldsymbol{s}\boldsymbol{A}_{\rho(i)} + \boldsymbol{e}_i \\
\hat{\boldsymbol{c}}_i &= M_{i,1}(\boldsymbol{s}\boldsymbol{y}^\top, \overbrace{0, \ldots, 0}^{m-1}) + \left[\sum_{j\in\{2,\ldots,s_{\max}\}} M_{i,j}\boldsymbol{v}_j\right] - \boldsymbol{s}\boldsymbol{H}_{\rho(i)} + \hat{\boldsymbol{e}}_i
\end{aligned}$$

and outputs

$$\mathsf{CT} = \left((\boldsymbol{M}, \rho), \{\boldsymbol{c}_i\}_{i\in[\ell]}, \{\hat{\boldsymbol{c}}_i\}_{i\in[\ell]}, C = \mathsf{MSB}(\boldsymbol{s}\boldsymbol{y}^\top) \oplus \mathsf{msg}\right).$$

Dec$(\mathsf{PK}, \mathsf{CT}, \mathsf{MSK})$**:** Decryption takes as input the public parameters PK, a ciphertext CT encrypting some message under some LSSS access policy $(\boldsymbol{M}, \rho)$ with the properties stipulated in Lemma 4.1, and the secret key SK corresponding to some subset of attributes $U \subseteq \mathbb{U}$. If $(1, 0, \ldots, 0)$ is *not* in the span of the rows of $\boldsymbol{M}$ associated with U, then decryption fails. Otherwise, let I be a set of row indices of the matrix $\boldsymbol{M}$ such that $\forall i \in I\colon \rho(i) \in U$ and let $\{w_i\}_{i\in I} \in \{0,1\} \subset \mathbb{Z}_q$ be scalars such that $\sum_{i\in I} w_i\boldsymbol{M}_i = (1, 0, \ldots, 0)$, where $\boldsymbol{M}_i$ is the i^{th} row of $\boldsymbol{M}$.

Note that the existence of such scalars $\{w_i\}_{i\in I}$ and their efficient determination for the LSSS generated by the algorithm from Sect. 4.2 are guaranteed by Lemma 4.1. The procedure computes

$$K' = \sum_{i\in I} w_i \left(\boldsymbol{c}_i \boldsymbol{k}_{\rho(i)}^\top + \hat{\boldsymbol{c}}_i \boldsymbol{t}^\top \right)$$

and outputs

$$\mathsf{msg}' = C \oplus \mathsf{MSB}(K').$$

Correctness: We show that the scheme is correct. Consider a set of attributes $U \subseteq \mathbb{U}$ and any LSSS access policy $(\boldsymbol{M}, \rho)$ for which U constitute an authorized set. By construction,

$$K' = \sum_{i\in I} w_i \left(\boldsymbol{c}_i \boldsymbol{k}_{\rho(i)}^\top + \hat{\boldsymbol{c}}_i \boldsymbol{t}^\top \right).$$

Expanding $\{\boldsymbol{c}_i\}_{i\in I}$ and $\{\hat{\boldsymbol{c}}_i\}_{i\in I}$, we get

$$K' = \sum_{i\in I} w_i \boldsymbol{s} \boldsymbol{A}_{\rho(i)} \boldsymbol{k}_{\rho(i)}^\top + \sum_{i\in I} w_i M_{i,1} (\boldsymbol{s}\boldsymbol{y}^\top, 0, \ldots, 0) \boldsymbol{t}^\top + \sum_{i\in I, j\in\{2,\ldots,s_{\max}\}} w_i M_{i,j} \boldsymbol{v}_j \boldsymbol{t}^\top - \sum_{i\in I} w_i \boldsymbol{s} \boldsymbol{H}_{\rho(i)} \boldsymbol{t}^\top + \sum_{i\in I} w_i \boldsymbol{e}_i \boldsymbol{k}_{\rho(i)}^\top + \sum_{i\in I} w_i \hat{\boldsymbol{e}}_i \boldsymbol{t}^\top.$$

Recall that for each $u \in U$, we have $\boldsymbol{k}_u = \hat{\boldsymbol{k}}_u + \tilde{\boldsymbol{k}}_u$ and $\boldsymbol{A}_u \tilde{\boldsymbol{k}}_u^\top = \boldsymbol{H}_u \boldsymbol{t}^\top - \boldsymbol{A}_u \hat{\boldsymbol{k}}_u^\top$. Therefore, for each $i \in I$, it holds that

$$\boldsymbol{A}_{\rho(i)} \boldsymbol{k}_{\rho(i)}^\top = \boldsymbol{A}_{\rho(i)} \hat{\boldsymbol{k}}_{\rho(i)}^\top + \boldsymbol{A}_{\rho(i)} \tilde{\boldsymbol{k}}_{\rho(i)}^\top = \boldsymbol{H}_{\rho(i)} \boldsymbol{t}^\top.$$

Hence,

$$\begin{aligned} K' &= \cancel{\sum_{i\in I} w_i \boldsymbol{s} \boldsymbol{H}_{\rho(i)} \boldsymbol{t}^\top} + \sum_{i\in I} w_i M_{i,1} (\boldsymbol{s}\boldsymbol{y}^\top, 0, \ldots, 0) \boldsymbol{t}^\top + \sum_{i\in I, j\in\{2,\ldots,s_{\max}\}} w_i M_{i,j} \boldsymbol{v}_j \boldsymbol{t}^\top - \cancel{\sum_{i\in I} w_i \boldsymbol{s} \boldsymbol{H}_{\rho(i)} \boldsymbol{t}^\top} + \sum_{i\in I} w_i \boldsymbol{e}_i \boldsymbol{k}_{\rho(i)}^\top + \sum_{i\in I} w_i \hat{\boldsymbol{e}}_i \boldsymbol{t}^\top \\ &= \sum_{i\in I} w_i M_{i,1} (\boldsymbol{s}\boldsymbol{y}^\top, 0, \ldots, 0) \boldsymbol{t}^\top + \sum_{i\in I, j\in\{2,\ldots,s_{\max}\}} w_i M_{i,j} \boldsymbol{v}_j \boldsymbol{t}^\top + \sum_{i\in I} w_i \boldsymbol{e}_i \boldsymbol{k}_{\rho(i)}^\top \\ &\quad + \sum_{i\in I} w_i \hat{\boldsymbol{e}}_i \boldsymbol{t}^\top \\ &= \left(\sum_{i\in I} w_i M_{i,1} \right) (\boldsymbol{s}\boldsymbol{y}^\top, 0, \ldots, 0) \boldsymbol{t}^\top + \sum_{j\in\{2,\ldots,s_{\max}\}} \left(\sum_{i\in I} w_i M_{i,j} \right) \boldsymbol{v}_j \boldsymbol{t}^\top \\ &\quad + \sum_{i\in I} w_i \boldsymbol{e}_i \boldsymbol{k}_{\rho(i)}^\top + \sum_{i\in I} w_i \hat{\boldsymbol{e}}_i \boldsymbol{t}^\top. \end{aligned}$$

Recall that $\sum_{i\in I} w_i M_{i,1} = 1$. Also, for $1 < j \leq s_{\max}$, $\sum_{i\in I} w_i M_{i,j} = 0$. Additionally, $\boldsymbol{t} = (1, \hat{\boldsymbol{t}})$, and hence, $(\boldsymbol{s}\boldsymbol{y}^\top, 0, \ldots, 0)\boldsymbol{t}^\top = \boldsymbol{s}\boldsymbol{y}^\top$. Thus,

$$K' = \boldsymbol{s}\boldsymbol{y}^\top + \sum_{i\in I} w_i \boldsymbol{e}_i \boldsymbol{k}_{\rho(i)}^\top + \sum_{i\in I} w_i \hat{\boldsymbol{e}}_i \boldsymbol{t}^\top.$$

Correctness now follows since the last two terms are small and should not affect the MSB of $\boldsymbol{s}\boldsymbol{y}^\top$. To see this, we observe that the following inequalities hold except with negligible probability:

- $\|\boldsymbol{e}_i\| \leq \sqrt{m}\sigma$: This follows directly from Lemma 3.3 since each of the m coordinates of $\boldsymbol{e}_i$ comes from the truncated discrete Gaussian distribution $\widetilde{\mathcal{D}}_{\mathbb{Z},\sigma}$.
- $\|\hat{\boldsymbol{e}}_i\| \leq \sqrt{m}\hat{B}$: This holds since each of the m coordinates of $\hat{\boldsymbol{e}}_i$ comes from the uniform distribution over $\mathbb{Z} \cap [-\hat{B}, \hat{B}]$.
- $\|\boldsymbol{k}_{\rho(i)}\| \leq m\sigma + \sqrt{m}\hat{B}$: This holds since $\boldsymbol{k}_{\rho(i)} = \hat{\boldsymbol{k}}_{\rho(i)} + \tilde{\boldsymbol{k}}_{\rho(i)}$, where (1) $\|\hat{\boldsymbol{k}}_{\rho(i)}\| \leq \sqrt{m}\hat{B}$ since each of its m coordinates comes from the uniform distribution over $\mathbb{Z} \cap [-\hat{B}, \hat{B}]$ and (2) $\|\tilde{\boldsymbol{k}}_{\rho(i)}\| \leq m\sigma$ since it comes from a distribution that is statistically close to the truncated discrete Gaussian distribution $\widetilde{\mathcal{D}}_{\mathbb{Z}^m,\sigma}$.
- $\|\boldsymbol{t}\| < m\sigma$: This holds since $\boldsymbol{t} = (1, \hat{\boldsymbol{t}})$, where $\hat{\boldsymbol{t}}$ comes from a truncated discrete Gaussian distribution $\widetilde{\mathcal{D}}_{\mathbb{Z}^{m-1},\sigma}$.

Using the fact that the w_i's are in $\{0, 1\}$ (Lemma 4.1), we have that

$$\begin{aligned}\|\sum_{i\in I} w_i \boldsymbol{e}_i \boldsymbol{k}_{\rho(i)}^\top + \sum_{i\in I} w_i \hat{\boldsymbol{e}}_i \boldsymbol{t}^\top\| &< |\mathbb{U}|\,(m^{3/2}\sigma^2 + m\sigma\hat{B} + m^{3/2}\sigma\hat{B}) \\ &< |\mathbb{U}| \cdot 3m^{3/2}\sigma\hat{B} < q/4,\end{aligned}$$

where the last inequality is by the parameter setting as shown above. Thus, with all but negligible probability in λ, the MSB of $\boldsymbol{s}\boldsymbol{y}^\top$ is not affected by the above noise which is bounded by $q/4$ and therefore does not affect the most significant bit. Namely, $\mathsf{MSB}(K') = \mathsf{MSB}(\boldsymbol{s}\boldsymbol{y}^\top)$. This completes the proof of correctness.

6 Our Multi-authority ABE Scheme

In this section, we present our MA-ABE scheme for access structures represented by DNF formulas. The scheme is associated with a universe of global identifiers $\mathcal{GID} \subset \{0,1\}^*$, a universe of authority identifiers $\mathcal{AU}$, and we will use the Lewko-Waters [42] transformation to represent the DNF access policies as monotone LSSS. More precisely, we only design an MA-ABE scheme for LSSS access policies $(\boldsymbol{M}, \rho)$ with properties stipulated in Lemma 4.1, that is, we construct an MA-ABE scheme for LSSS access policies $(\boldsymbol{M}, \rho)$ such that the entries of $\boldsymbol{M}$ come from $\{-1, 0, 1\}$ as well as reconstruction only involves coefficients coming from $\{0, 1\}$, and prove the scheme to be statically secure under linear

independence restriction (see [25, Definition 3.7] in the full version for the formal description of the security model). Thanks to the observation made by [1] as mentioned in Remark 4.1, our MA-ABE scheme actually achieves the standard notion of static security (see [25, Definition 3.6] in the full version for the formal description of the security model) when implemented for the class of all access structures represented by DNF formulas. We will assume each authority controls only one attribute in our scheme. However, it can be readily generalized to a scheme where each authority controls an a priori bounded number of attributes using standard techniques [42]. Further, we will assume that all access policies $(\boldsymbol{M}, \rho)$ used in our scheme correspond to a matrix $\boldsymbol{M}$ with at most $s_{\max}$ columns and an injective row-labeling function ρ, i.e., an authority/attribute is associated with at most one row of $\boldsymbol{M}$. Since the Lewko-Waters transformation [42] introduces a new column for the resulting LSSS matrix for each AND gate in the underlying formula, the bound in the number of columns of the LSSS matrices naturally translates to the number of AND gates of the supported DNF formulas at implementation. Similar to our CP-ABE scheme, in our scheme description below, we assume for simplicity of presentation that both the encryption and the decryption algorithms receive an access policy directly in its LSSS representation. However, we note that in the actual implementation, the encryption and decryption algorithms should instead take in the DNF representation of the access policy and deterministically compute its LSSS representation using the Lewko-Waters transformation algorithm [42].

First, we provide the parameter constraints required by our correctness and security proof. Fix any $0 < \epsilon < 1/2$. For any $B \in \mathbb{N}$, let $\mathcal{U}_B$ denote the uniform distribution on $\mathbb{Z} \cap [-B, B]$, i.e., integers between $\pm B$. The Setup algorithm chooses parameters n, m, σ, q and noise distributions $\chi_{\mathsf{lwe}}, \chi_1, \chi_2, \chi_{\mathsf{big}}$, satisfying the following constraints:

- $n = \mathsf{poly}(\lambda)$, $\sigma < q$, $n \cdot q/\sigma < 2^{n^\epsilon}$, $\chi_{\mathsf{lwe}} = \widetilde{\mathcal{D}}_{\mathbb{Z},\sigma}$ (for LWE security)
- $m > 2s_{\max} n \log q + \omega \log n + 2\lambda$ (for enhanced trapdoor sampling and LHL)
- $\sigma > \sqrt{s_{\max} n \log q \log m} + \lambda$ (for enhanced trapdoor sampling)
- $\chi_1 = \widetilde{\mathcal{D}}_{\mathbb{Z}^{m-1},\sigma}$, $\chi_2 = \widetilde{\mathcal{D}}_{\mathbb{Z}^m,\sigma}$ (for enhanced trapdoor sampling)
- $\chi_{\mathsf{big}} = \mathcal{U}_{\hat{B}}$, where $\hat{B} > m^{3/2}\sigma 2^\lambda$ (for smudging/security)
- $|\mathcal{AU}|\,(m^{3/2}\sigma^2 + 2m\hat{B}^2) < q/4$ (for correctness)

We will now describe our MA-ABE construction.

GlobalSetup$(1^\lambda, s_{\max})$: The global setup algorithm takes in the security parameter λ encoded in unary and the maximum width $s_{\max} = s_{\max}(\lambda)$ of an LSSS matrix supported by the scheme. It first chooses an LWE modulus q, dimensions n, m, and also distributions $\chi_{\mathsf{lwe}}, \chi_1, \chi_2, \chi_{\mathsf{big}}$ as described above. Next, it samples a vector $\boldsymbol{y} \leftarrow \mathbb{Z}_q^n$ and sets the matrix $\boldsymbol{B}_1 \in \mathbb{Z}_q^{n \times m}$ as $\boldsymbol{B}_1 = \left[\boldsymbol{y}^\top \| \overbrace{\boldsymbol{0}^\top \| \cdots \| \boldsymbol{0}^\top}^{m-1}\right]$, where each $\boldsymbol{0} \in \mathbb{Z}_q^n$. Furthermore, we assume a hash function $\mathsf{H}\colon \mathcal{GID} \rightarrow (\mathbb{Z} \cap [-\hat{B}, \hat{B}])^{m-1}$ mapping strings $\mathsf{GID} \in \mathcal{GID}$ to random $(m-1)$-dimensional

vectors of integers in the interval $[-\hat{B}, \hat{B}]$. H will be modeled as a random oracle in the security proof. Finally, it outputs the hash function H and the global parameters

$$\mathsf{GP} = (n, m, q, s_{\max}, \chi_{\mathsf{lwe}}, \chi_1, \chi_2, \chi_{\mathsf{big}}, \boldsymbol{B}_1).$$

$\mathsf{AuthSetup}(\mathsf{GP}, \mathsf{H}, u)$: Given the global parameters GP, the hash function H, and an authority identifier $u \in \mathcal{AU}$, the algorithm generates a matrix-trapdoor pair $(\boldsymbol{A}_u, T_{\boldsymbol{A}_u}) \leftarrow \mathsf{EnTrapGen}(1^n, 1^m, q)$ such that $\boldsymbol{A}_u \in \mathbb{Z}_q^{n \times m}$, samples another matrix $\boldsymbol{H}_u \leftarrow \mathbb{Z}_q^{n \times m}$, and outputs the pair of public key and secret key for the authority u

$$\mathsf{PK}_u = (\boldsymbol{A}_u, \boldsymbol{H}_u), \quad \mathsf{MSK}_u = T_{\boldsymbol{A}_u}.$$

$\mathsf{KeyGen}(\mathsf{GP}, \mathsf{H}, \mathsf{GID}, \mathsf{MSK}_u)$: The key generation algorithm takes as input the global parameters GP, the hash function H, the user's global identifier GID, and the authority's secret key MSK_u. It first computes the vector $\boldsymbol{t}_{\mathsf{GID}} = (1, \mathsf{H}(\mathsf{GID})) \in \mathbb{Z}^m$. Next, it chooses a vector $\hat{\boldsymbol{k}}_{\mathsf{GID},u} \leftarrow \chi_{\mathsf{big}}^m$, samples a vector $\tilde{\boldsymbol{k}}_{\mathsf{GID},u} \leftarrow \mathsf{EnSamplePre}(\boldsymbol{A}_u, T_{\boldsymbol{A}_u}, \sigma, \boldsymbol{t}_{\mathsf{GID}} \boldsymbol{H}_u^\top - \hat{\boldsymbol{k}}_{\mathsf{GID},u} \boldsymbol{A}_u^\top)$, and outputs the secret key for the user GID as

$$\mathsf{SK}_{\mathsf{GID},u} = \hat{\boldsymbol{k}}_{\mathsf{GID},u} + \tilde{\boldsymbol{k}}_{\mathsf{GID},u}.$$

$\mathsf{Enc}(\mathsf{GP}, \mathsf{H}, \mathsf{msg}, (\boldsymbol{M}, \boldsymbol{\rho}), \{\mathsf{PK}_u\})$: The encryption algorithm takes as input the global parameters GP, the hash function H, a message bit $\mathsf{msg} \in \{0, 1\}$ to encrypt, an LSSS access policy $(\boldsymbol{M}, \rho)$ generated by the Lewko-Waters transformation [42], where $\boldsymbol{M} = (M_{i,j})_{\ell \times s_{\max}} \in \{-1, 0, 1\}^{\ell \times s_{\max}} \subset \mathbb{Z}_q^{\ell \times s_{\max}}$ (Lemma 4.1) and $\rho \colon [\ell] \to \mathcal{AU}$, and public keys of the relevant authorities $\{\mathsf{PK}_u\}$. The function ρ associates rows of $\boldsymbol{M}$ to authorities (recall that we assume that each authority controls a single attribute). We assume that ρ is an injective function. The procedure samples vectors $\boldsymbol{s} \leftarrow \mathbb{Z}_q^n, \{\boldsymbol{v}_j\}_{j \in \{2, \ldots, s_{\max}\}} \leftarrow \mathbb{Z}_q^m$, and $\{\boldsymbol{x}_i\}_{i \in [\ell]} \leftarrow \mathbb{Z}_q^n$. It additionally samples vectors $\{\boldsymbol{e}_i\}_{i \in [\ell]} \leftarrow \chi_{\mathsf{lwe}}^m$ and $\{\hat{\boldsymbol{e}}_i\}_{i \in [\ell]} \leftarrow \chi_{\mathsf{big}}^m$. For each $i \in [\ell]$, it computes vectors $\boldsymbol{c}_i, \hat{\boldsymbol{c}}_i \in \mathbb{Z}_q^m$ as follows:

$$\boldsymbol{c}_i = \boldsymbol{x}_i \boldsymbol{A}_{\rho(i)} + \boldsymbol{e}_i$$

$$\hat{\boldsymbol{c}}_i = M_{i,1} \boldsymbol{s} \boldsymbol{B}_1 + \left[\sum_{j \in \{2, \ldots, s_{\max}\}} M_{i,j} \boldsymbol{v}_j \right] - \boldsymbol{x}_i \boldsymbol{H}_{\rho(i)} + \hat{\boldsymbol{e}}_i$$

and outputs

$$\mathsf{CT} = \left((\boldsymbol{M}, \rho), \{\boldsymbol{c}_i\}_{i \in [\ell]}, \{\hat{\boldsymbol{c}}_i\}_{i \in [\ell]}, C = \mathsf{MSB}(\boldsymbol{s} \boldsymbol{y}^\top) \oplus \mathsf{msg} \right).$$

$\mathsf{Dec}(\mathsf{GP}, \mathsf{H}, \mathsf{CT}, \mathsf{GID}, \{\mathsf{SK}_{\mathsf{GID},u}\})$: Decryption takes as input the global parameters GP, the hash function H, a ciphertext CT generated with respect to an LSSS

access policy $(\boldsymbol{M}, \rho)$ generated by the Lewko-Waters transformation [42], a user identity GID, and the secret keys $\left\{\mathsf{SK}_{\mathsf{GID},\rho(i)}\right\}_{i\in I}$ corresponding to a subset I of row indices of the access matrix $\boldsymbol{M}$ possessed by that user. If $(1, 0, \ldots, 0)$ is *not* in the span of the rows of $\boldsymbol{M}$ having indices in the set I, then decryption fails. Otherwise, let $\{w_i\}_{i\in I} \in \{0, 1\} \subset \mathbb{Z}_q$ be scalars such that $\sum_{i\in I} w_i \boldsymbol{M}_i = (1, 0, \ldots, 0)$, where $\boldsymbol{M}_i$ is the ith row of $\boldsymbol{M}$. The existence of such scalars $\{w_i\}_{i\in I}$ and their efficient determination are guaranteed by [1,14,42]. The algorithm computes the vector $\boldsymbol{t}_{\mathsf{GID}} = (1, \mathsf{H}(\mathsf{GID})) \in \mathbb{Z}^m$ followed by

$$K' = \sum_{i\in I} w_i \cdot \left(\boldsymbol{c}_i \mathsf{SK}_{\mathsf{GID},\rho(i)}^{\top} + \hat{\boldsymbol{c}}_i \boldsymbol{t}_{\mathsf{GID}}^{\top}\right),$$

and outputs

$$\mathsf{msg}' = C \oplus \mathsf{MSB}(K').$$

References

1. Agrawal, S., Biswas, R., Nishimaki, R., Xagawa, K., Xie, X., Yamada, S.: Cryptanalysis of Boyen's attribute-based encryption scheme in TCC 2013 (2020, private communication)
2. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28
3. Agrawal, S., Boneh, D., Boyen, X.: Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 98–115. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_6
4. Agrawal, S., Freeman, D.M., Vaikuntanathan, V.: Functional encryption for inner product predicates from learning with errors. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 21–40. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_2
5. Agrawal, S., Maitra, M., Yamada, S.: Attribute based encryption (and more) for nondeterministic finite automata from LWE. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 765–797. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_26
6. Agrawal, S., Wichs, D., Yamada, S.: Optimal broadcast encryption from LWE and pairings in the standard model. In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12550, pp. 149–178. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64375-1_6
7. Agrawal, S., Yamada, S.: CP-ABE for circuits (and more) in the symmetric key setting. In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12550, pp. 117–148. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64375-1_5
8. Agrawal, S., Yamada, S.: Optimal broadcast encryption from pairings and LWE. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12105, pp. 13–43. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45721-1_2
9. Ajtai, M.: Generating hard instances of the short basis problem. In: Wiedermann, J., van Emde Boas, P., Nielsen, M. (eds.) ICALP 1999. LNCS, vol. 1644, pp. 1–9. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48523-6_1

10. Asharov, G., Jain, A., Wichs, D.: Multiparty computation with low communication, computation and interaction via threshold FHE. IACR Cryptology ePrint Archive 2011/613 (2011)
11. Attrapadung, N.: Unbounded dynamic predicate compositions in attribute-based encryption. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11476, pp. 34–67. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17653-2_2
12. Benaloh, J., Leichter, J.: Generalized secret sharing and monotone functions. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 27–35. Springer, New York (1990). https://doi.org/10.1007/0-387-34799-2_3
13. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: S&P 2007, pp. 321–334. IEEE (2007)
14. Boneh, D., et al.: Threshold cryptosystems from threshold fully homomorphic encryption. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10991, pp. 565–596. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_19
15. Boneh, D., et al.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_30
16. Boyen, X.: Attribute-based functional encryption on lattices. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 122–142. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36594-2_8
17. Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: STOC 2013, pp. 575–584. ACM (2013)
18. Brakerski, Z., Vaikuntanathan, V.: Lattice-inspired broadcast encryption and succinct ciphertext-policy ABE. IACR Cryptology ePrint Archive 2020/191 (2020)
19. Brakerski, Z., Vaikuntanathan, V.: Circuit-ABE from LWE: unbounded attributes and semi-adaptive security. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9816, pp. 363–384. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53015-3_13
20. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_27
21. Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70936-7_28
22. Chase, M., Chow, S.S.M.: Improving privacy and security in multi-authority attribute-based encryption. In: CCS 2009, pp. 121–130. ACM (2009)
23. Chen, J., Gay, R., Wee, H.: Improved dual system ABE in prime-order groups via predicate encodings. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 595–624. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_20
24. Chen, J., Gong, J., Kowalczyk, L., Wee, H.: Unbounded ABE via bilinear entropy expansion, revisited. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10820, pp. 503–534. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78381-9_19
25. Datta, P., Komargodski, I., Waters, B.: Decentralized multi-authority ABE for DNFs from LWE. IACR Cryptology ePrint Archive 2020/1386 (2020)
26. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976)

27. Garg, S., Gentry, C., Halevi, S., Sahai, A., Waters, B.: Attribute-based encryption for circuits from multilinear maps. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 479–499. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_27
28. Gentry, C., Gorbunov, S., Halevi, S.: Graph-induced multilinear maps from lattices. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9015, pp. 498–527. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_20
29. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC 2008, pp. 197–206. ACM (2008)
30. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_5
31. Gong, J., Waters, B., Wee, H.: ABE for DFA from k-Lin. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 732–764. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_25
32. Gong, J., Wee, H.: Adaptively secure ABE for DFA from k-Lin and more. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12107, pp. 278–308. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_10
33. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: STOC 2013, pp. 545–554. ACM (2013)
34. Gorbunov, S., Vinayagamurthy, D.: Riding on asymmetry: efficient ABE for branching programs. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9452, pp. 550–574. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48797-6_23
35. Goyal, R., Koppula, V., Waters, B.: Collusion resistant traitor tracing from learning with errors. In: STOC 2018, pp. 660–670. ACM (2018)
36. Goyal, R., Koppula, V., Waters, B.: Lockable obfuscation. In: FOCS 2017, pp. 612–621. IEEE (2017)
37. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: CCS 2006, pp. 89–98. ACM (2006)
38. Karchmer, M., Wigderson, A.: On span programs. In: Structure in Complexity Theory Conference 1993, pp. 102–111. IEEE (1993)
39. Kim, S.: Multi-authority attribute-based encryption from LWE in the OT model. IACR Cryptology ePrint Archive 2019/280 (2019)
40. Kowalczyk, L., Wee, H.: Compact adaptively secure ABE for NC^1 from k-Lin. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11476, pp. 3–33. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17653-2_1
41. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_4
42. Lewko, A., Waters, B.: Decentralizing attribute-based encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 568–588. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_31
43. Lewko, A., Waters, B.: Unbounded HIBE and attribute-based encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 547–567. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_30

44. Lin, H., Cao, Z., Liang, X., Shao, J.: Secure threshold multi authority attribute based encryption without a central authority. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT 2008. LNCS, vol. 5365, pp. 426–436. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89754-5_33
45. Micciancio, D., Peikert, C.: Trapdoors for lattices: simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_41
46. Micciancio, D., Peikert, C.: Hardness of SIS and LWE with small parameters. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 21–39. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_2
47. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. SIAM J. Comput. **37**(1), 267–302 (2007)
48. Müller, S., Katzenbeisser, S., Eckert, C.: Distributed attribute-based encryption. In: Lee, P.J., Cheon, J.H. (eds.) ICISC 2008. LNCS, vol. 5461, pp. 20–36. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00730-9_2
49. Müller, S., Katzenbeisser, S., Eckert, C.: On multi-authority ciphertext-policy attribute-based encryption. Bull. Korean Math. Soc. **46**, 803–819 (2009)
50. Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: CCS 2007, pp. 195–203. ACM (2007)
51. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: STOC 2009, pp. 333–342. ACM (2009)
52. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC 2005, pp. 84–93. ACM (2005)
53. Rouselakis, Y., Waters, B.: Efficient statically-secure large-universe multi-authority attribute-based encryption. In: Böhme, R., Okamoto, T. (eds.) FC 2015. LNCS, vol. 8975, pp. 315–332. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47854-7_19
54. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_27
55. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: FOCS 1994, pp. 124–134. IEEE (1994)
56. Tsabary, R.: Fully secure attribute-based encryption for t-CNF from LWE. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11692, pp. 62–85. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26948-7_3
57. Wang, Z., Fan, X., Liu, F.-H.: FE for inner products and its application to decentralized ABE. In: Lin, D., Sako, K. (eds.) PKC 2019. LNCS, vol. 11443, pp. 97–127. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17259-6_4
58. Waters, B.: Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 53–70. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19379-8_4