Multi-Client Functional Encryption with Fine-Grained Access Control

Ky Nguyen[1], Duong Hieu Phan[2], and David Pointcheval[1]

[1] DIENS, École normale supérieure, CNRS, Inria, PSL University, Paris, France
[2] LTCI, Telecom Paris, Institut Polytechnique de Paris, France

Abstract. Multi-Client Functional Encryption (MCFE) and Multi-Input Functional Encryption (MIFE) are very interesting extensions of Functional Encryption for practical purpose. They allow to compute joint function over data from multiple parties. Both primitives are aimed at applications in multi-user settings where decryption can be correctly output for users with appropriate functional decryption keys only. While the definitions for a single user or multiple users were quite general and can be realized for general classes of functions as expressive as Turing machines or all circuits, efficient schemes have been proposed so far for concrete classes of functions: either only for access control, *i.e.* the identity function under some conditions, or linear/quadratic functions under no condition.

In this paper, we target classes of functions that explicitly combine some evaluation functions independent of the decrypting user under the condition of some access control. More precisely, we introduce a framework for MCFE with fine-grained access control and propose constructions for both single-client and multi-client settings, for inner-product evaluation and access control via Linear Secret Sharing Schemes (LSSS), with selective and adaptive security. The only known work that combines functional encryption in multi-user setting with access control was proposed by Abdalla *et al.* (Asiacrypt '20), which relies on a generic transformation from the single-client schemes to obtain MIFE schemes that suffer a quadratic factor of n (where n denotes the number of clients) in the ciphertext size. We follow a different path, via MCFE: we present a *duplicate-and-compress* technique to transform the single-client scheme and obtain a MCFE with fine-grained access control scheme with only a linear factor of n in the ciphertext size. Our final scheme thus outperforms the Abdalla *et al.*'s scheme by a factor n, as one can obtain MIFE from MCFE by making all the labels in MCFE a fixed public constant. The concrete constructions are secure under the SXDH assumption, in the random oracle model for the MCFE scheme, but in the standard model for the MIFE improvement.

1 Introduction

Encryption enables people to securely communicate and share sensitive data in an *all-or-nothing* fashion: once the recipients have the secret key then they will recover the original data, otherwise the recipients have no information about

the plaintext data. Functional Encryption (FE) [40, 17], introduced by Boneh, Sahai and Waters, overcomes this all-or-nothing limitation of PKE by allowing recipients to recover encrypted data in a more fine-grained manner: instead of revealing the whole original encrypted data, recipients can get the result of evaluation of some function on the data, according to the function associated to the decryption key, called *functional decryption key*. By allowing computation of partial data, one can aim at getting both: the utility of analysis on large data while preserving personal information private.

FE received large interest from the cryptographic community, first as a generalization of Identity-Based Encryption (IBE) [42, 23, 15, 16] and Attribute-Based Encryption (ABE) [40, 29, 38, 9, 37], which are unfortunately only access control, with all-or-nothing decryption as a result. Abdalla *et al.* [2] proposed the first construction for evaluating a concrete function: the inner product between a vector in the ciphertext and a vector in the functional decryption key, hence coined IPFE. The interest in FE then increased, especially in the multi-user setting in which the inputs come from different users, possibly in competition, and the output characterizes a joint function on the inputs [20, 32]. Applications are then numerous, and the encryptors can even be the final recipients of aggregated results. Then, this might look similar to multi-party computation (MPC), where several players privately provide their inputs to allow computations on them. But the main difference is that functional encryption is expected as a non-interactive process, and thus quite more interesting in practice. While FE with a single encryptor might be of theoretical interest, in real-life, the number of really useful functions may be limited. When this number of functions is small, any PKE can be converted into FE by additionally encrypting the evaluations by the various functions under specific keys. This approach is impossible for multiple users, even when a unique fixed function is considered.

In the multi-user case, Goldwasser *et al.* [27, 28] introduced the notion of Multi-Input Functional Encryption (MIFE) and Multi-Client Functional Encryption (MCFE) where the single input x to the encryption procedure is broken down into an input vector $(x_1, \ldots, x_n)$ with independent components. An index i for each client and, in the case of MCFE, a (typically time-based) tag tag are used for every encryption: $(c_1 = \mathsf{Enc}(1, x_1, \mathsf{tag}), \ldots, c_n = \mathsf{Enc}(n, x_n, \mathsf{tag}))$. Anyone owning a functional decryption key dk_f, for an n-ary function f and multiple ciphertexts (for the same tag tag, in the case of MCFE) can compute $f(x_1, \ldots, x_n)$ but nothing else about the individual x_i's. Implicitly, clients have to be able to coordinate together on the tags, and different usability in practice. In particular, in MCFE, the combination of ciphertexts generated for different tags does not give a valid global ciphertext and the adversary learns nothing from it. This leads to more versatility since encrypting x_i under tag has a different meaning from encrypting x_i under $\mathsf{tag}' \neq \mathsf{tag}$. On the other hand, MIFE does not use tags and once a ciphertext of x_i is computed, it can be reused for different combinations. However, in both situations, encryption must require a private key, otherwise anybody could complete the vector initiated by a user in many ways, and then obtain many various evaluations from a unique functional decryption key. But then, since

encryption needs a private key per user, for each component c_i, some of these keys might get corrupted. Therefore, there are two main distinguishing aspects regarding MCFE that have to be dealt with: the role of tags in construction and the danger of corruption for security.

Another classical issue with encryption is the decryption key, even if legitimately obtained: once delivered, it can be used forever. One may expect revocation, or access control with more fine-grained authentication. This has been extensively studied with broadcast encryption, revocation systems and more generally, with attribute-based encryption (ABE) [44]. Finally, as already explained, FE is a generalization of IBE and ABE, and after having been illustrated with IBE and ABE, linear evaluations [6, 3, 14, 18] and quadratic evaluations [10, 26, 8, 33] have been proposed. However, there are still very few works that combine function evaluation and access control with concrete schemes. This could provide FE, with concrete function evaluation for some target users, or revocation (of users or functions). Abdalla *et al.* [4] have been the first to address this problem, for enhancing FE and MIFE with access control. In addition, they informally argue that from an ABE for MIFE one can lift it for free to get MCFE, thus solving both problems at the same time. Precisely, they mentioned "*by resorting for instance, to the notion of multi-client IPFE, where ciphertexts are associated with time-stamps, and only ciphertext with matching time-stamps can be combined (e.g. [20]) we believe that our proposed primitive provides a more general and versatile solution to the problem*". Their idea can be interpreted as: tags can be used as specific attributes, and tags can be embedded in policies to automatically obtain multi-client settings. This argument seems formally valid when considering the general form of MIFE and MCFE. However, when considering concrete classes of functions, which is our main focus in this paper, it is unlikely to be efficiently feasible and we will explain the reason in the technical overview in Section 3. We underline that the principal difference between MCFE and MIFE is the presence of tags for producing the ciphertext components, which can be jointly decrypted only if all tags are equal. Thus, we can retrieve an MIFE from MCFE by fixing and publishing one tag, which retains the *same* ciphertext's size from the MCFE scheme to the new MIFE one. Moreover, since the combination of ciphertext components in MCFE is restrained by the tags, its security model is far less restrictive than the security model of MIFE that has to deal with arbitrary combination of ciphertext components. For these reasons, our main objective becomes constructing an MCFE having smaller ciphertext size while permitting access control over decryption keys.

We take a completely different approach than in [4] to answer this question. Borrowing the terminology from ABE, our work will focus on *key-policy* (KP) constructions, where the policy is defined at the moment of key extraction and a ciphertext associated with certain attributes can be decrypted only if those attributes satisfy the policy. The dual notion of *ciphertext-policy* (CP) constructions is already studied in [4]. We concentrate solely on particular functionality classes whose description contains two separate parts: a description of functions exclusively for evaluation and a binary relation exclusively for

modeling access control. Although this conceptual point of view does *not* take us out of the FE realm and thus can be captured by the general FE notion, it suits perfectly our purpose to compute inner-products along with fine-grained access control provided by Linear Secret Sharing Schemes (LSSS) in this paper. Then, we start from single-client IPFE schemes with LSSS access control and leverage them to get an MCFE scheme, where only tags are needed for hashing during encryption, and the hash function is modeled as a random oracle. Removing labels by fixing a public tag for all ciphertexts leads to an MIFE scheme in the standard model that is more efficient than the one from [4].

1.1 Related Work

Recently, [30] improves upon the single-client construction based on Learning with Errors (LWE) from [4], for IPFE with access control expressed by bounded depth boolean circuits, achieving better security along with smaller ciphertexts. In another work, [39] also studies LWE-based single client constructions for IPFE with access control expressed by general boolean functions but under selective challenge attributes. The single-client LWE-based construction in [39] is later lifted to an MIFE using the generic transformation from [4].

Also in the single-client setting, another line of works attempts to construct FE for a general uniform functionality class such as Turing machines (TMFE), which naturally captures inner-product evaluation under LSSS access control. The work of Agrawal *et al.* [7] provided a non-adaptively simulation-based secure construction for TMFE in the *dynamic bounded collusion* model under sub-exponential LWE. The construction is later improved in [5] to achieve adaptive security under polynomial LWE, DDH or bilinear decisional Diffie-Hellman in specific groups, or quadratic residuosity. Towards this goal, both works of [7, 5] additionally gave constructions of FE for circuits of *unbounded* size and depth, which can also encompass inner-product computation under LSSS access control, based on various standard assumptions such as computational Diffie-Hellman, factoring, or polynomial LWE. All single-client constructions from [7, 5] use a wide range of cryptographic primitives in a generic manner, which deviates from our goal to give explicit constructions in the multi-user setting.

1.2 Our Contributions

Single-client setting. We propose new single-client schemes whose selectively-secure version is almost as efficient as the selectively-secure version in [4] and the adaptively-secure version is nearly three times as efficient as the adaptively-secure version in [4]. More importantly, our schemes can be extended to multi-client settings. Our constructions exploit the *Dual Pairing Vector Spaces* proposed by Okamoto-Takashima [35, 37].

Multi-client setting. Our main contribution is an extension from single-client to multi-client without linearly increasing the complexity in the number n of clients. The generic transformation proposed by Abdalla *et al.* [4, Theorem 6.3]

Scheme	$\mathcal{P}$	$\mathcal{F}$	$\lvert\mathsf{ct}\rvert$	**Security**
[4, Sect. 3.1]	MSP; CP	$\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{MSP}}$	$n+2d+2$	sel-sim
[4, Sect. 3.2]	roMSP; CP	$\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{roMSP}}$	$3nd+3d+2$	ad-ind
Sect. 4, Fig. 1	LSSS; KP	$\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{LSSS}}$	$n+8d+4$	sel-ind
	LSSS; KP		$nd+2n+7d+3$	ad-ind
[4, Sect. 6.2] applied to [4, Sect. 3.1]	MSP; CP	$\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{MSP}}$	$n^2+2nd+2n$	mi-ad-ind
Sect. 5.2	LSSS; KP	$\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{LSSS}}$	$8nd+5n$	mc-ad-ind

Table 1: We compare our constructions with existing works, in terms of the number of group elements in the ciphertext (column $\lvert\mathsf{ct}\rvert$), the largest predicate class that can be handled (column $\mathcal{P}$), the function class (column $\mathcal{F}$), security (column **Security**). We denote by d the number of attributes needed by the policy in a ciphertext. All our schemes are defined for the functionality class $\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{LSSS}} = \mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$ constituted by $\mathcal{F}^{\mathsf{IP}} = \{F_{\mathbf{y}} : \mathbb{Z}_q^n \to \mathbb{Z}_q; \mathbf{x} \mapsto \langle \mathbf{x}, \mathbf{y} \rangle \in \mathcal{R}(\mathbb{Z}_q)\}$ and LSSS of Linear Secret Sharing Schemes over attributes in $\mathbb{Z}_q$, where $n, q \in \mathbb{N}$, q is prime and $|\mathcal{R}(\mathbb{Z}_q)| = \mathsf{poly}(\log q)$. The schemes from [4] are constructed for $\mathcal{F}^{\mathsf{IP}} \times \mathsf{MSP}$ and $\mathcal{F}^{\mathsf{IP}} \times \mathsf{roMSP}$, where $\mathsf{MSP}, \mathsf{roMSP}$ are classes of *monotone span programs, read-once monotone span programs* over attributes in $\mathbb{Z}_q$. The shorthands $(\mathsf{mc}, \mathsf{mi}, \mathsf{sel}, \mathsf{ad}, \mathsf{ind}, \mathsf{sim})$ denote multi-client setting, multi-input setting, selective security, adaptive security, indistinguishability-based, simulation-based.

results in a degradation of factor n in both construction and security reduction. As previously stated, Abdalla *et al.*'s generic transformation can only help to achieve a multi-input scheme and is unlikely to be generalized to a multi-client scheme without further seriously degrading efficiency. On the other hand, because MIFE can be defined as MCFE with a fixed public constant tag, our construction yields a much more efficient MIFE with access control than the Abdalla *et al.*'s scheme (in fact, n times more efficient). More concretely, the total communication among n clients in our MCFE construction is a linear function in n and does not suffer a quadratic blow-up of n^2 group elements as in [4].

Comparisons. Our concrete constructions focus on the functionality class whose member's description contains inner-product evaluation functions and binary relations to describe access control. In the pairing-based setting, we give comparisons with existing works in Table 1. Recall that in MCFE, n can be a large number of clients, while d is the number of attributes, generally small, used in a policy. Concretely, we can consider identity-based functional encryption, as outlined in [4], where $d = 1$, whatever the size of n: our ciphertext's size is linear instead of quadratic in n as in [4].

Organization. We first give the necessary preliminaries in Section 2, then we present the high-level ideas and intuitions of our results in Section 3, before

going into purely technical details in Section 4 for the single-client schemes and in Section 5 for the multi-client schemes.

2 Preliminaries

We write $[n]$ to denote the set $\{1, 2, \ldots, n\}$ for an integer n. For any $q \geq 2$, we let $\mathbb{Z}_q$ denote the ring of integers with addition and multiplication modulo q. For a prime q and an integer N, we denote by $GL_N(\mathbb{Z}_q)$ the general linear group of of degree N over $\mathbb{Z}_q$. We write vectors as row-vectors, unless stated otherwise. For a vector $\mathbf{x}$ of dimension n, the notation $\mathbf{x}[i]$ indicates the i-th coordinate of $\mathbf{x}$, for $i \in [n]$. We will follow the implicit notation in [25] and use $[\![a]\!]$ to denote g^a in a cyclic group $\mathbb{G}$ of prime order q generated by g, given $a \in \mathbb{Z}_q$. This implicit notation extends to matrices and vectors having entries in $\mathbb{Z}_q$. We use the shorthand ppt for "probabilistic polynomial time". In the security proofs, whenever we use an ordered sequence of games $(\mathsf{G}_0, \mathsf{G}_1, \ldots, \mathsf{G}_i, \ldots, \mathsf{G}_L)$ indexed by $i \in \{0, 1, \ldots, L\}$, we refer to the predecessor of G_j by G_{j-1}, for $j \in [L]$.

2.1 Hardness Assumptions

We state the assumptions needed for our constructions.

Definition 1. *In a cyclic group $\mathbb{G}$ of prime order q, the* **Decisional Diffie-Hellman** (DDH) *problem is to distinguish the distributions*

$$D_0 = \{([\![1]\!], [\![a]\!], [\![b]\!], [\![ab]\!])\} \qquad D_1 = \{([\![1]\!], [\![a]\!], [\![b]\!], [\![c]\!])\}.$$

for $a, b, c \xleftarrow{\$} \mathbb{Z}_q$. The DDH *assumption in $\mathbb{G}$ assumes that no* ppt *adversary can solve the* DDH *problem with non-negligible probability.*

Definition 2. *In the bilinear setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$, the* **Symmetric eXternal Diffie-Hellman** (SXDH) *assumption makes the* DDH *assumption in both $\mathbb{G}_1$ and $\mathbb{G}_2$.*

2.2 Dual Pairing Vector Spaces

Our constructions rely on the *Dual Pairing Vector Spaces* (DPVS) framework in prime-order bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ and $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}$ are all written additively. The DPVS technique dates back to the seminal work by Okamoto-Takashima [35, 36, 37] aiming at adaptive security for ABE as well as IBE, together with the *dual system methodology* introduced by Waters [43]. In [31], the setting for dual systems is composite-order bilinear groups. Continuing on this line of works, Chen *et al.* [19] used prime-order bilinear groups under the SXDH assumption. Let us fix $N \in \mathbb{N}$ and consider $\mathbb{G}_1^N$ having N copies of $\mathbb{G}_1$. Any $\mathbf{x} = [\![(x_1, \ldots, x_N)]\!]_1 \in \mathbb{G}_1^N$ is identified as the vector $(x_1, \ldots, x_N) \in \mathbb{Z}_q^N$. There is no ambiguity because $\mathbb{G}_1$ is a cyclic group of order q prime. The $\mathbf{0}$-vector is $\mathbf{0} = [\![(0, \ldots, 0)]\!]_1$. The addition of two vectors in $\mathbb{G}_1^N$ is defined

by coordinate-wise addition. The scalar multiplication of a vector is defined by $t \cdot \mathbf{x} := [\![t \cdot (x_1, \ldots, x_N)]\!]_1$, where $t \in \mathbb{Z}_q$ and $\mathbf{x} = [\![(x_1, \ldots, x_N)]\!]_1$. The additive inverse of $\mathbf{x} \in \mathbb{G}_1^N$ is defined to be $-\mathbf{x} := [\![(-x_1, \ldots, -x_N)]\!]_1$. Viewing $\mathbb{Z}_q^N$ as a vector space of dimension N over $\mathbb{Z}_q$ with the notions of bases, we can obtain naturally a similar notion of bases for $\mathbb{G}_1^N$. More specifically, any invertible matrix $B \in GL_N(\mathbb{Z}_q)$ identifies a basis $\mathbf{B}$ of $\mathbb{G}_1^N$, whose i-th row $\mathbf{b}_i$ is $[\![B^{(i)}]\!]_1$, where $B^{(i)}$ is the i-th row of B. The canonical basis $\mathbf{A}$ of $\mathbb{G}_1^N$ consists of $\mathbf{a}_1 := [\![(1, 0 \ldots, 0)]\!]_1, \mathbf{a}_2 := [\![(0, 1, 0 \ldots, 0)]\!]_1, \ldots, \mathbf{a}_N := [\![(0, \ldots, 0, 1)]\!]_1$. It is straightforward that we can write $\mathbf{B} = B \cdot \mathbf{A}$ for any basis $\mathbf{B}$ of $\mathbb{G}_1^N$ corresponding to an invertible matrix $B \in GL_N(\mathbb{Z}_q)$. We write $\mathbf{x} = (x_1, \ldots, x_N)_{\mathbf{B}}$ to indicate the representation of $\mathbf{x}$ in the basis $\mathbf{B}$, i.e. $\mathbf{x} = \sum_{i=1}^N x_i \cdot \mathbf{b}_i$. By convention the writing $\mathbf{x} = (x_1, \ldots, x_N)$ concerns the canonical basis $\mathbf{A}$.

Treating $\mathbb{G}_2^N$ similarly, we can furthermore define a product of two vectors $\mathbf{x} = [\![(x_1, \ldots, x_N)]\!]_1 \in \mathbb{G}_1^N, \mathbf{y} = [\![(y_1, \ldots, y_N)]\!]_2 \in \mathbb{G}_2^N$ by $\mathbf{x} \times \mathbf{y} := \prod_{i=1}^N \mathbf{e}(\mathbf{x}[i], \mathbf{y}[i]) = [\![\langle (x_1, \ldots, x_N), (y_1, \ldots, y_N) \rangle]\!]_{\mathsf{t}}$. Given a basis $\mathbf{B} = (\mathbf{b}_i)_{i \in [N]}$ of $\mathbb{G}_1^N$, we define $\mathbf{B}^*$ to be a basis of $\mathbb{G}_2^N$ by first defining $B' := (B^{-1})^\top$ and the i-th row $\mathbf{b}_i^*$ of $\mathbf{B}^*$ is $[\![B'^{(i)}]\!]_2$. It holds that $B \cdot (B')^\top = I_N$ the identity matrix and $\mathbf{b}_i \times \mathbf{b}_j^* = [\![\delta_{i,j}]\!]_{\mathsf{t}}$ for every $i, j \in [N]$, where $\delta_{i,j} = 1$ if and only if $i = j$. We call the pair $(\mathbf{B}, \mathbf{B}^*)$ a *pair of dual orthogonal bases* of $(\mathbb{G}_1^N, \mathbb{G}_2^N)$. If $\mathbf{B}$ is constructed by a random invertible matrix $B \xleftarrow{\$} GL_N(\mathbb{Z}_q)$, we call the resulting $(\mathbf{B}, \mathbf{B}^*)$ a pair of random dual bases. A DPVS is a bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_{\mathsf{t}}, g_1, g_2, g_{\mathsf{t}}, \mathbf{e}, q, N)$ with dual orthogonal bases. In this work, we also use extensively *basis changes* over dual orthogonal bases of a DPVS to argue the steps of switching key as well as ciphertext vectors to semi-functional mode in our proofs. The details of such basis changes are recalled in the full version [34].

2.3 Access Structure and Linear Secret Sharing Schemes

We recall below the vocabularies of access structures and linear secret sharing schemes that will be used in this work. Let $\mathsf{Att} = \{\mathsf{att}_1, \mathsf{att}_2, \ldots, \mathsf{att}_m\}$ be a finite universe of attributes. An *access structure* over Att is a family $\mathbb{A} \subseteq 2^{\mathsf{Att}} \setminus \{\varnothing\}$. A set in $\mathbb{A}$ is said to be *authorized*; otherwise it is *unauthorized*. An access structure $\mathbb{A}$ is *monotone* if $\mathsf{S}_1 \subseteq \mathsf{S}_2 \subseteq \mathsf{Att}$ and $\mathsf{S}_1 \in \mathbb{A}$ imply $\mathsf{S}_2 \in \mathbb{A}$. Given a set of attributes $\mathsf{S} \subseteq \mathsf{Att}$, we write $\mathbb{A}(\mathsf{S}) = 1$ if and only if there exists $A \subseteq \mathsf{S}$ such that A is authorized. A secret sharing scheme for an access structure $\mathbb{A}$ over the attributes $\mathsf{Att} = \{\mathsf{att}_1, \mathsf{att}_2, \ldots, \mathsf{att}_m\}$ allows sharing a secret s among the m attributes att_j for $1 \leq j \leq m$, such that: (1) Any authorized set in $\mathbb{A}$ can be used to reconstruct s from the shares of its elements; (2) Given any unauthorized set and its shares, the secret s is statistically identical to a uniform random value. We will use *linear secret sharing schemes* (LSSS), which is recalled below:

Definition 3 (LSSS [11]). *Let K be a field, $d, f \in \mathbb{N}$, and Att be a finite universe of attributes. A* Linear Secret Sharing Scheme LSSS *over K for an access structure $\mathbb{A}$ over Att is specified by a share-generating matrix $\mathbf{A} \in K^{d \times f}$ such that for any $I \subset [d]$, there exists a vector $\mathbf{c} \in K^d$ with support I and $\mathbf{c} \cdot \mathbf{A} = (1, 0, \ldots, 0)$ if and only if $\{\mathsf{att}_i \mid i \in I\} \in \mathbb{A}$.*

In order to share s using an LSSS over K, one first picks uniformly random values $v_2, v_3, \ldots, v_f \xleftarrow{\$} K$ and the share for an attribute att_i is the i-th coordinate $\mathbf{s}[i]$ of the share vector $\mathbf{s} \coloneqq (s, v_2, v_3, \ldots, v_f) \cdot \mathbf{A}^\top$. Then, only an authorized set $\{\mathsf{att}_i \mid i \in I\} \in \mathbb{A}$ for some $I \subseteq [d]$ can recover $\mathbf{c}$ to reconstruct s from the shares by: $\mathbf{c} \cdot \mathbf{s}^\top = \mathbf{c} \cdot (\mathbf{A} \cdot (s, v_2, v_3, \ldots, v_f)^\top) = s$. Some canonical examples of LSSS include Shamir's secret sharing scheme for any f-out-of-d threshold gate [41] or Benaloh and Leichter's scheme for any monotone formula [13]. An access structure $\mathbb{A}$ is said to be LSSS-*realizable* if there exists a linear secret sharing scheme implementing $\mathbb{A}$.

Let $y \in \mathbb{Z}_q$ where q is prime and for the sake of simplicity, let $\mathsf{Att} \subset \mathbb{Z}_q$ be a set of attributes. Let $\mathbb{A}$ be a monotone access structure over Att realizable by an LSSS over $\mathbb{Z}_q$. A *random labeling* procedure $\Lambda_y(\mathbb{A})$ is a secret sharing of y using LSSS:

$$\Lambda_y(\mathbb{A}) \coloneqq (y, v_2, v_3, \ldots, v_f) \cdot \mathbf{A}^\top \in \mathbb{Z}_q^d \tag{1}$$

where $\mathbf{A} \in \mathbb{Z}_q^{d \times f}$ is the share-generating matrix and $v_2, v_3, \ldots, v_f \xleftarrow{\$} \mathbb{Z}_q$.

2.4 The Masking Lemma

We state a technical lemma that is employed throughout our proofs. A detailed proof can be found in the full version [34]. The general purposes of the variables τ, x, y, z_j in the lemma are discussed in the technical overview in Section 3.2.

Lemma 4 (Adapted from [35, 36, 37, 24]). *Let $\mathbb{A}$ be an LSSS-realizable over a set of attributes $\mathsf{Att} \subseteq \mathbb{Z}_q$. We denote by $\mathsf{List\text{-}Att}(\mathbb{A})$ the list of attributes appearing in $\mathbb{A}$ and by P the cardinality of $\mathsf{List\text{-}Att}(\mathbb{A})$. Let $\mathsf{S} \subseteq \mathsf{Att}$ be a set of attributes. Let $(\mathbf{H}, \mathbf{H}^*)$ and $(\mathbf{F}, \mathbf{F}^*)$ be two random dual bases of $(\mathbb{G}_1^2, \mathbb{G}_2^2)$ and $(\mathbb{G}_1^8, \mathbb{G}_2^8)$, respectively. The vectors $(\mathbf{h}_1, \mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3)$ are public, while all other vectors are secret. Suppose we have two random labelings $(a_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})} \leftarrow \Lambda_{a_0}(\mathbb{A})$ and $(a'_j)_j \leftarrow \Lambda_{a'_0}(\mathbb{A})$ for $a_0, a'_0 \xleftarrow{\$} \mathbb{Z}_q$. Then, under the SXDH assumption in $(\mathbb{G}_1, \mathbb{G}_2)$, the following two distributions are computationally indistinguishable:*

$$D_1 \coloneqq \left\{ \begin{array}{ll} x, y & \\ \forall\, j \in \mathsf{S}: & \mathbf{c}_j = (\sigma_j \cdot (1, -j),\ \psi,\ 0,\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}} \\ \forall\, j \in \mathsf{List\text{-}Att}(\mathbb{A}): & \mathbf{k}^*_j = (\pi_j \cdot (j, 1),\ a_j \cdot z,\ 0,\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}^*} \\ & \mathbf{c}_{\mathsf{root}} = (\psi,\ 0)_{\mathbf{H}} \\ & \mathbf{k}^*_{\mathsf{root}} = (a_0 \cdot z,\ 0)_{\mathbf{H}^*} \end{array} \right\}$$

and

$$D_2 \coloneqq \left\{ \begin{array}{ll} x, y & \\ \forall\, j \in \mathsf{S}: & \mathbf{c}_j = (\sigma_j \cdot (1, -j),\ \psi,\ 0,\ 0,\ \tau z_j \cdot x,\ 0,\ 0)_{\mathbf{F}} \\ \forall\, j \in \mathsf{List\text{-}Att}(\mathbb{A}): & \mathbf{k}^*_j = (\pi_j \cdot (j, 1),\ a_j \cdot z,\ 0,\ 0,\ a'_j \cdot y / z_j,\ 0,\ 0)_{\mathbf{F}^*} \\ & \mathbf{c}_{\mathsf{root}} = (\psi,\ \tau \cdot x)_{\mathbf{H}} \\ & \mathbf{k}^*_{\mathsf{root}} = (a_0 \cdot z,\ a'_0 \cdot y)_{\mathbf{H}^*} \end{array} \right\}$$

for any $x, y \in \mathbb{Z}_q$ and $z_j, \sigma_j, \pi_j, \psi, \tau, z, r'_0 \xleftarrow{\$} \mathbb{Z}_q$.

2.5 Functional Encryption with Fine-Grained Access Control

We first present the syntax of functional encryption with a fine-grained access control following the works in [4, 30, 39]. The functionality class is $\mathcal{F} \times \mathsf{AC\text{-}K}$. The evaluation functions is taken from $\mathcal{F} := \{F_\lambda : \mathcal{D}_\lambda \to \mathcal{R}_\lambda\}_\lambda$ is a family of functions indexed by security parameters $\lambda \in \mathbb{N}$. When $F_\lambda, \mathcal{D}_\lambda$, and $\mathcal{R}_\lambda$ are clear from context, we drop the subscript λ and use the shorthands $F, \mathcal{D}$, and $\mathcal{R}$ respectively. The access control is captured by a relation $\mathsf{Rel} : \mathsf{AC\text{-}K} \times \mathsf{AC\text{-}Ct} \to \{0,1\}$, for some sets $\mathsf{AC\text{-}Ct}$ and $\mathsf{AC\text{-}K}$. A plaintext consists of $(\mathsf{ac\text{-}ct}, x) \in \mathsf{AC\text{-}Ct} \times \mathcal{D}_\lambda$, whose corresponding ciphertext can be decrypted to $F_\lambda(x)$ using the functional key $\mathsf{sk}_{F_\lambda,\mathsf{ac\text{-}k}}$ for $\mathsf{ac\text{-}k} \in \mathsf{AC\text{-}K}$ if and only if $\mathsf{Rel}(\mathsf{ac\text{-}k}, \mathsf{ac\text{-}ct}) = 1$. The syntax of such functional encryption schemes is given below:

Definition 5 (Functional encryption with fine-grained access control). *A* functional encryption scheme with fine-grained access control *for* $\mathcal{F} \times \mathsf{AC\text{-}K}$ *consists of four algorithms* $(\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$*:*

$\mathsf{Setup}(1^\lambda)$**:** *Given as input a security parameter* λ*, output a pair* $(\mathsf{pk}, \mathsf{msk})$.
$\mathsf{Extract}(\mathsf{msk}, F_\lambda, \mathsf{ac\text{-}k})$**:** *Given* $\mathsf{ac\text{-}k} \in \mathsf{AC\text{-}K}$*, a function description* $F_\lambda \in \mathcal{F}$*, and the master secret key* msk*, output a secret key* $\mathsf{sk}_{F_\lambda,\mathsf{ac\text{-}k}}$.
$\mathsf{Enc}(\mathsf{pk}, x, \mathsf{ac\text{-}ct})$**:** *Given as inputs* $\mathsf{ac\text{-}ct} \in \mathsf{AC\text{-}Ct}$*, the public key* pk*, and a message* $x \in \mathcal{D}_\lambda$*, output a ciphertext* ct.
$\mathsf{Dec}(\mathsf{sk}_{F_\lambda,\mathsf{ac\text{-}k}}, \mathsf{ct})$**:** *Given the functional secret key* $\mathsf{sk}_{F_\lambda,\mathsf{ac\text{-}k}}$*, and a ciphertext* ct*, output an element in* $\mathcal{R}_\lambda$ *or an invalid symbol* $\perp$.

Correctness. For sufficiently large $\lambda \in \mathbb{N}$, for all $(F_\lambda, \mathsf{ac\text{-}k}) \in \mathcal{F} \times \mathsf{AC\text{-}K}$ and $(\mathsf{msk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(1^\lambda), \mathsf{sk}_{F_\lambda,\mathsf{ac\text{-}k}} \leftarrow \mathsf{Extract}(\mathsf{msk}, F_\lambda, \mathsf{ac\text{-}k})$ for all $\mathsf{ac\text{-}ct}$ satisfying $\mathsf{Rel}(\mathsf{ac\text{-}k}, \mathsf{ac\text{-}ct}) = 1$, it holds with overwhelming probability that

$$\mathsf{Dec}(\mathsf{sk}_{F_\lambda,\mathsf{ac\text{-}k}}, \mathsf{Enc}(\mathsf{pk}, x, \mathsf{ac\text{-}ct})) = F_\lambda(x) \text{ whenever } F_\lambda(x) \neq \perp^{3} \ ,$$

where the probability is taken over the random coins of the algorithms.

Security. We recall in the full version [34] the notion of *indistinguishability-based security against chosen-plaintext attacks (IND-CPA)* in the same manner as in [2], taking into account the attribute-based control using policies, as well as a *simulation-based* notion in a selective setting as in [4].

Remark 6. In Sections 4 and 5, our concrete constructions instantiate $\mathsf{AC\text{-}K}$ as a class of policies and $\mathsf{AC\text{-}Ct}$ as a superset over an attribute space, while the relation is the natural evaluation $\mathsf{Rel}(\mathbb{A} \in \mathsf{AC\text{-}K}, \mathsf{S} \in \mathsf{AC\text{-}Ct}) := \mathbb{A}(\mathsf{S})$. Following the terminology of ABE schemes, our constructions are *key-policy* (KP). By treating $\mathsf{AC\text{-}K}$ as a superset over an attribute space and $\mathsf{AC\text{-}Ct}$ as a class of policies, we will obtain *ciphertext-policy* (CP) schemes. The KP and CP notions are symmetric in terms of how we determine the support $\mathsf{AC\text{-}K} \times \mathsf{AC\text{-}Ct}$ of Rel.

[3] See [12, 1] for discussions about this relaxation. The general reason is that some functionality might contain $\perp$ in its range and if $F_\lambda(x) = \perp$ we do not impose $\mathsf{Dec}(\mathsf{sk}_{F_\lambda,\mathsf{ac\text{-}k}}, \mathsf{Enc}(\mathsf{pk}, x, \mathsf{ac\text{-}ct})) = F_\lambda(x)$, neither do we disallow it.

3 Technical Overview

3.1 Formalizing Access Control in Functional Encryption

First of all, we discuss how we formalize access control in the notion of functional encryption, which will affect our formal definitions in both single-client setting (Definition 5) and multi-client setting (Definition 8). On the one hand, accompanying an encryption scheme with access control over decryption keys is already expressed by ABE, which in itself is a special case of FE. Thus, FE schemes with fine-grained access control can be described by the general FE notion for any class of functions that can handle the desired access control along with the required computation.

On the other hand, when working with concrete functionality, we usually find ourselves in the context where the evaluation *cannot* express the access control and they cannot be described abstractly using a single functionality. Therefore, in this paper we consider FE with access control as FE schemes for *particular* functionality class whose description can be separated into two parts $\mathcal{F} \times \mathsf{AC\text{-}K}$: (1) a first part $F \in \mathcal{F}$ for evaluation, (2) and a second part for access control captured by a binary relation $\mathsf{Rel} : \mathsf{AC\text{-}K} \times \mathsf{AC\text{-}Ct} \to \{0,1\}$, for some sets $\mathsf{AC\text{-}K}, \mathsf{AC\text{-}Ct}$. The key extraction is done with respect to $(\mathsf{ac\text{-}k} \in \mathsf{AC\text{-}K}, F)$, meanwhile the encryption procedure will receive $(\mathsf{ac\text{-}ct} \in \mathsf{AC\text{-}Ct}, x)$. A key $\mathsf{sk}_{\mathsf{ac\text{-}k},F}$ can decrypt a ciphertext $\mathsf{ct}_{\mathsf{ac\text{-}ct}}(x)$ to $F(x)$ if and only if $\mathsf{Rel}(\mathsf{ac\text{-}k}, \mathsf{ac\text{-}ct}) = 1$. We stress that this way of formulation does not take us out of the FE regime, as it is still captured by the general FE notion.

We show how the above fomalization is used in a concrete case. In the following discussion we will distinguish the $\boxed{\text{input}}$ during encryption from the $\ulcorner\text{parameters}\urcorner$ during key extraction. In this paper we focus on $F \in \mathcal{F}^{\mathsf{IP}} = \{F_{\mathbf{y}} : \mathbb{Z}_q^n \to \mathbb{Z}_q\}$ for computing inner products over $\mathbb{Z}_q^n$ for some prime q and $n \in \mathbb{N}$, where $F_{\mathbf{y}}(\mathbf{x}) := \langle \mathbf{x}, \mathbf{y} \rangle$. The simplest non-trivial example for access control is identity-based control, i.e. $\mathsf{AC\text{-}K} = \mathsf{AC\text{-}Ct} = \mathsf{ID}$ for some identity space ID and $\mathsf{Rel}_{\mathsf{ibe}}(\mathsf{id\text{-}k}, \mathsf{id\text{-}ct}) = \left(\mathsf{id\text{-}k} \stackrel{?}{=} \mathsf{id\text{-}ct}\right)$. The functional keys are extracted using $\ulcorner(\mathsf{id\text{-}k}, \mathbf{y})\urcorner$ and the ciphertexts are encrypted using $\boxed{(\mathsf{id\text{-}ct}, \mathbf{x})}$. First of all, it is *not* immediate how $\mathcal{F}^{\mathsf{IP}}$ can be used to implement the check $\tau z \cdot (\llcorner \mathsf{id\text{-}k} \lrcorner - \boxed{\mathsf{id\text{-}ct}})$ for the identity-based control, where τ and z are random values generated for encryption and key extraction, respectively, together acting as a mask of the decryption value. Notably, the value z *cannot* be specified as part of the inner-product evaluation function, because the inner-product evaluation itself must be independent of users at the time of generating functional keys, *nor* as part of the ciphertext. It thus seems indispensable to treat the functionality as $\mathcal{F}^{\mathsf{IP}} \times \mathsf{ID}$: the functional key is generated w.r.t $F_{\ulcorner \mathbf{y} \urcorner} \in \mathcal{F}^{\mathsf{IP}}$ and $\llcorner \mathsf{id\text{-}k} \lrcorner \in \mathsf{ID}$, while the ciphertext is encrypted w.r.t $\boxed{(\mathsf{id\text{-}ct} \in \mathsf{ID}, \mathbf{x} \in \mathbb{Z}_q^n)}$. During decryption for obtaining $\langle \boxed{\mathbf{x}}, \ulcorner \mathbf{y} \urcorner \rangle + \tau z \cdot (\llcorner \mathsf{id\text{-}k} \lrcorner - \boxed{\mathsf{id\text{-}ct}})$, the ID-part of the functional key will implement the control $\tau z \cdot (\llcorner \mathsf{id\text{-}k} \lrcorner - \boxed{\mathsf{id\text{-}ct}})$ whilst the $\mathcal{F}^{\mathsf{IP}}$-part will compute $\langle \boxed{\mathbf{x}}, \ulcorner \mathbf{y} \urcorner \rangle$.

Treatment of Tags in MCFE with Access Control. As mentioned in the introduction, our current objective is constructing MCFE schemes with access

control having smaller ciphertexts. We use the functionality $\mathcal{F}^{\mathsf{IP}} \times \mathsf{ID}$ as a running example. The input $\boxed{\mathbf{x}}$ for inner-product calculation is broken down into n components for the entries x_i of $\boxed{\mathbf{x}}$. The encryption procedure takes $\boxed{(x_i, \mathsf{id\text{-}ct}_i, \mathsf{tag}_i)}$ and outputs a ciphertext component ct_i, for some identity $\boxed{\mathsf{id\text{-}ct}_i}$ and a tag $\boxed{\mathsf{tag}_i}$. The decryption procedure receives a functional key, which is derived from $F_{\lceil \mathbf{y} \rfloor} \in \mathcal{F}^{\mathsf{IP}}$ and $\lceil \bar{\mathsf{id}}\text{-}\bar{\mathsf{k}} \rfloor \in \mathsf{ID}$, and the n ciphertext components $(\mathsf{ct}_i)_{i=1}^n$. The decrypted result is $\langle \boxed{\mathbf{x}}, \lceil \mathbf{y} \rfloor \rangle$ if $\boxed{\mathsf{id\text{-}ct}_i} = \lceil \bar{\mathsf{id}}\text{-}\bar{\mathsf{k}} \rfloor$ for all i and $\boxed{\mathsf{tag}_i} = \boxed{\mathsf{tag}_j}$ for all i, j. In the setting that the identities and tags can be public, if the identity control does not pass or if the tags are not the same, a totally random value is returned by the decryption procedure. We now face the same problem of checking equality among $\boxed{\mathsf{tag}_i}$ in the same manner that has to be done for identities from ID.

First of all, it is unlikely that we want to embed the checks $\boxed{\mathsf{tag}_i} \stackrel{?}{=} \boxed{\mathsf{tag}_j}$ in the $\mathcal{F}^{\mathsf{IP}}$-part. More specifically, we would have to make the decryption compute $(\sum_{i=1}^n \boxed{x_i}, \lceil \bar{y}_i \rfloor) + \tau z \cdot (\lceil \bar{\mathsf{id}}\text{-}\bar{\mathsf{k}} \rfloor - \boxed{\mathsf{id\text{-}ct}_i}) + \sum_{i=1}^{n-1} z_i(\boxed{\mathsf{tag}_i} - \boxed{\mathsf{tag}_{i+1}})$ from n ciphertext components ct_i of $\boxed{(x_i, \mathsf{id\text{-}ct}_i)}$, for some random values $z, z_i \xleftarrow{\$} \mathbb{Z}_q$ and $\lceil \mathbf{y} = (y_1, ..., y_n) \rfloor$. It is worth noting that the check $z_i(\boxed{\mathsf{tag}_i} - \boxed{\mathsf{tag}_{i+1}})$ needs two values defined at encryption time and not key extraction time. Therefore, in order for the functional key to "perform" the n required checks, all n tags $\boxed{(\mathsf{tag}_1, \ldots, \mathsf{tag}_n)}$ must be encrypted in an IBE-style in ct_i. Roughly speaking, this makes each ct_i of size linear in n, due to the number of group elements required for encrypting the n tags, in addition to a constant number of group elements for encrypting $\boxed{(x_i, \mathsf{id\text{-}ct}_i)}$. Thus the total communication increases to quadratic in n over all n components ct_i, which is exactly what we are trying to avoid.

Furthermore, it might be tempting to embed the equality checks in the access control but because $\boxed{\mathsf{tag}_i, \mathsf{tag}_j}$ are defined only at encryption time, they are unknown to the key extraction for the ID-part. More generally, in a setting that permits a *different* [4] attribute set $\boxed{\mathsf{S}_i}$ in each individual ciphertext, one can try to regard $\boxed{\mathsf{tag}_i}$ as an attribute in $\boxed{\mathsf{S}_i}$. The correctness insists on the condition $\lceil \mathbb{A} \rfloor(\boxed{\mathsf{S}_i}) = 1$ for all i and the equality checks $\boxed{\mathsf{tag}_i} \stackrel{?}{=} \boxed{\mathsf{tag}_j}$ must somehow be done by $\lceil \mathbb{A} \rfloor(\boxed{\mathsf{S}_i})$, which is not possible due to the fact that $\boxed{\mathsf{tag}_j}$ is independent of both $\lceil \mathbb{A} \rfloor$ and $\boxed{\mathsf{S}_i}$. Consequently, we have to cope with the tags independently from the functionality's description. As a final remark, this also demonstrates the gap between MIFE and MCFE for the concrete functionality to compute inner products under access control by access structures, even though the general notion of MIFE can describe MCFE, provided that the evaluation functions of the underlying functionality class can test equality between $\boxed{\mathsf{tag}_i}$.

[4] If all clients must use the *same* set of attributes $\boxed{\mathsf{S}}$, we can treat $\boxed{\mathsf{tag}_i}$ as a virtual attribute in S, while enforcing the same $\boxed{\mathsf{S}}$ for all i. This implies that all $\boxed{\mathsf{tag}_i}$ must be the same. However, this approach requires a consensus among all n clients on S, which general might be more complicated than agreeing on tag.

3.2 Adaptively Secure Single-Client Construction

Our construction for functional encryption schemes with fine-grained access control is using *Dual Pairing Vector Spaces* (DPVSes). We highlight our main ideas to achieve adaptive security. We refer to Section 2.2 for background on DPVSes. Our schemes are key-policy, such that the access structure $\mathbb{A}$ is expressed in the key using vectors $\{(\mathbf{k}^*_j)_{j\in\mathsf{List\text{-}Att}(\mathbb{A})}, \mathbf{k}_{\mathsf{root}}\}$ over $\mathbb{G}_2$ and a set S of attributes are embedded in the ciphertext using vectors $\{(\mathbf{c}_j)_{j\in\mathsf{S}}, \mathbf{c}_{\mathsf{root}}\}$ over $\mathbb{G}_1$, where $\mathsf{List\text{-}Att}(\mathbb{A})$ is the list of attributes appearing in the access structure $\mathbb{A}$. We use a linear secret sharing scheme based on $\mathbb{A}$ to create the shares $(a_j)_{j\in\mathsf{List\text{-}Att}(\mathbb{A})}$ of $a_0 \xleftarrow{\$} \mathbb{Z}_q$. The shares will then be embedded in the functional secret key components $(\mathbf{k}^*_j)_{j\in\mathsf{List\text{-}Att}(\mathbb{A})}$. When all the components corresponding to an authorized set in $\mathbb{A}$ are present, the shares can be combined to reconstruct the secret value a_0, which is now embedded in a key component $\mathbf{k}^*_{\mathsf{root}}$. In all vectors $(\mathbf{c}_j)_j$ and $\mathbf{c}_{\mathsf{root}}$, we put a random value ψ. Intuitively, $[\![\psi a_0]\!]_{\mathsf{t}}$ is masking the IPFE-related ciphertext of Agrawal *et al.*'s type [6]. The vectors $((\mathbf{k}^*_j)_{j\in\mathsf{List\text{-}Att}(\mathbb{A})}, \mathbf{k}^*_{\mathsf{root}})$ and $((\mathbf{c}_j)_{j\in\mathsf{S}}, \mathbf{c}_{\mathsf{root}})$ lie in the dual orthogonal bases. Performing the products $\mathbf{c}_j \times \mathbf{k}^*_j$ and combining over $j \in \mathsf{S}$, where S is an authorized set, will permit recovering $[\![\psi a_0]\!]_{\mathsf{t}}$ that can be used to cancel out $[\![\psi a_0]\!]_{\mathsf{t}}$ in $\mathbf{c}_{\mathsf{root}} \times \mathbf{k}_{\mathsf{root}}$:

$$\begin{array}{cccc|c|c|c|c} \mathbf{c}_j & (& \cdots & \psi & 0 & \cdots &)_{\mathbf{F}} \; ; & \mathbf{c}_{\mathsf{root}} \; (\; \cdots \; | \; \psi \; | \; 0 \; | \; \cdots \;)_{\mathbf{H}} \\ \mathbf{k}^*_j & (& \cdots & a_j & 0 & \cdots &)_{\mathbf{F}^*}; & \mathbf{k}^*_{\mathsf{root}} \; (\; \cdots \; | \; a_0 \; | \; 0 \; | \; \cdots \;)_{\mathbf{H}^*} \end{array}$$

We use the techniques for adaptively-secure ABE introduced in the original work of Okamoto and Takashima [35, 36, 37] in the ensuing steps. In vein of the *dual-system methodology*, there are two modes of operation for keys and ciphertexts: a normal mode and a *semi-functional* mode. A normal key can decrypt any ciphertext, a semi-functional key can decrypt only normal ciphertexts, and decrypting semi-functional ciphertexts using semi-functional keys gives totally random values. The dual-system method proves security by a sequence of indistinguishable changes to make the challenge ciphertext semi-functional, then to make the keys semi-functional and in the end the challenge message will be perfectly hidden from the adversary. Interestingly, there is a twist stemming from the security model when integrating this technique into our security proofs for FE with access control: an adversary can additionally query for keys that work with the challenge ciphertext, i.e. the key's policy is satisfied. So as to achieve adaptive security, we have to be much more careful about which key to turn semi-functional, because the keys whose policies are satisfied should be capable of decrypting the (semi-functional) challenge ciphertext.

Our goal is to mask the value a_0 in $\mathbf{k}^*_{\mathsf{root}}$ by introducing a random mask $a'_0 y$ in the coordinate of *hidden* basis vectors, i.e. those that are not used at all in real life and are defined only for the proof, while the facing coordinate in $\mathbf{c}_{\mathsf{root}}$ is also changed to τx so as to mask ψ:

$$\begin{array}{cccc|c|c|c|c} \mathbf{c}_j & (& \cdots & \psi & \tau x z_j & \cdots &)_{\mathbf{F}} \; ; & \mathbf{c}_{\mathsf{root}} \; (\; \cdots \; | \; \psi \; | \; \tau x \; | \; \cdots \;)_{\mathbf{H}} \\ \mathbf{k}^*_j & (& \cdots & a_j & a'_j y / z_j & \cdots &)_{\mathbf{F}^*}; & \mathbf{k}^*_{\mathsf{root}} \; (\; \cdots \; | \; a_0 \; | \; a'_0 y \; | \; \cdots \;)_{\mathbf{H}^*} \end{array} .$$

The values x, y are known constants, $\tau, a'_0, (z_j)_j \xleftarrow{\$} \mathbb{Z}_q$, and $(a'_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}$ is another ensemble of secret shares for a'_0. Consequently, this will introduce a value $[\![\tau a'_0 xy]\!]_{\mathsf{t}}$ masking $[\![\psi a_0]\!]_{\mathsf{t}}$ when performing the product $\mathbf{c}_{\mathsf{root}} \times \mathbf{k}_{\mathsf{root}}$. We note that the value a'_0 is related to $(a'_j/z_j)_j$ by $a'_0 = \sum_{j \in \mathsf{S}'} z_j \cdot (a'_j/z_j)$ for any S' such that $\mathbb{A}(\mathsf{S}') = 1$. In the end, if $\mathbb{A}(\mathsf{S}) = 1$, from $\mathbf{c}_j$ and $\mathbf{k}_j$ it is possible to reconstruct $[\![\tau a'_0 xy]\!]_{\mathsf{t}}$ and recover $[\![\psi a_0]\!]_{\mathsf{t}}$. Otherwise, the entropy of a'_0 is preserved thanks to the randomness provided by $z_j \xleftarrow{\$} \mathbb{Z}_q$ for randomizing $(a'_j)_j$ to $(a'_j/z_j)_j$ in the components $(\mathbf{c}_j)_j$ of the *unique* challenge ciphertext[5], as well as the fact that $\mathbb{A}(\mathsf{S}) = 0$ means there will be some a'_j/z_j missing in the components $(\mathbf{k}^*_j)_j$ and the value z_j is information-theoretically hidden. Hence, if $\mathbb{A}(\mathsf{S}) = 0$ we will be able to change a'_0 to an independent and uniformly random value $r_0 \xleftarrow{\$} \mathbb{Z}^*_q$. It is obligatory that we apply this argument *key by key*, while considering the key's capability to decrypt the challenge ciphertext, because two different keys might mutually leak information about the same z_j and our statistically argument no longer holds. After a sequence of hybrids on the functional key queries, we can mask all the keys as desired so that the key and the challenge ciphertext will become readily semi-functional for later steps in the proof.

However, only for functional keys whose policy is not satisfied can we perform such a change from a'_0 to r_0, and we can decide the satisfiability only when the adversary adaptively queries for functional keys. Our idea is to introduce r_0 in *all* key components and at the same time use a mechanism to "cancel out" the masks $((a'_j/z_j)_j, r_0)$ in $((\mathbf{k}^*_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}, \mathbf{k}^*_{\mathsf{root}})$ if $\mathbb{A}(\mathsf{S}) = 1$. It is indispensable to have this mechanism because otherwise, as soon as we change a'_0 to r_0, even the reconstruction $\sum_{j \in \mathsf{S}'} z_j \cdot (a'_j/z_j) = a'_0$ is not able to remove r_0 for a correct decryption. In our particular setting for computing inner-products, we observe that if $\mathbb{A}(\mathsf{S}) = 1$, then $\langle \Delta\mathbf{x}, \mathbf{y} \rangle = 0$ for the sake of avoiding trivial attacks, where $\Delta\mathbf{x} \coloneqq \mathbf{x}^*_1 - \mathbf{x}^*_0$ is the difference of the two left-or-right challenge messages and $\mathbf{y}$ is specified the functional key. In the selective setting where $\Delta\mathbf{x}$ is known in advance, the key and ciphertext components can simply be masked using the constants $(x, y) \coloneqq (1, \langle \Delta\mathbf{x}, \mathbf{y} \rangle)$. However, for the goal of adaptive security where $\Delta\mathbf{x}$ is unknown at the time of key extraction, we have to make a trade-off and use DPVSes of dimensions linear in the dimension n of vectors for inner-products and mask the key and ciphertext components as follows:

$$
\begin{array}{c|ccc|c|c|c|c|c}
\mathbf{c}_j & (& \cdots & \psi & \tau z_j \Delta\mathbf{x}[1] & \cdots & \tau z_j \Delta\mathbf{x}[n] & \cdots &)_{\mathbf{F}} \\
\mathbf{k}^*_j & (& \cdots & a_j & a'_j \mathbf{y}[1]/z_j & \cdots & a'_j \mathbf{y}[n]/z_j & \cdots &)_{\mathbf{F}^*} \\
\hline
\mathbf{c}_{\mathsf{root}} & (& \cdots & \psi & \tau \Delta\mathbf{x}[1] & \cdots & \tau \Delta\mathbf{x}[n] & \cdots &)_{\mathbf{H}} \\
\mathbf{k}^*_{\mathsf{root}} & (& \cdots & a_0 & r_0 \mathbf{y}[1] & \cdots & r_0 \mathbf{y}[n] & \cdots &)_{\mathbf{H}^*}
\end{array}
$$

where each i-th pair of constants (x, y) is set to $(\Delta\mathbf{x}[i], \mathbf{y}[i])$ for all $i \in [n]$. Our arguments resort to a slight variant of the technique in [35, 36, 37], stated as a technical lemma (see Lemma 4) in Section 2.4. The lemma will use some auxiliary hidden vectors (which we do not show here) during the masking process and so as to economize the dimensions of our DPVSes, we apply the lemma n

[5] Since our single-client scheme is public-key, we can obtain multi-challenge security using a standard hybrid argument.

times in a sequence of hybrids to introduce $(\tau\Delta\mathbf{x}[i], r_0\mathbf{y}[i])_i$ while reusing and cleaning those auxiliary hidden vectors after each application. After successfully laying $(r_0\mathbf{y}[i])_i$ in place, the rest of the proof will use r_0 as a source of randomness to completely hide the challenge message. Our single-client constructions are presented in Section 4.

3.3 The "Duplicate-and-Compress" Technique

We give a glimpse of our main technical method to obtain a multi-client construction from our single-client construction, while maintaining the total ciphertext's size of order linear in n. The intriguing point we observe is as long as each client uses an independent DPVS, the technique we use to take care of the ciphertext/key vectors in the single-client case can be carried out in a *parallel* manner, to some extent. Therefore, in the security proof, we can distribute and accumulate in parallel the necessary information in small-dimension vectors rather than centralizing such information in few vectors of big dimension. Our treatment for the multi-client setting is twofold and we give below the main technical ideas.

The more restrictive MCFE. Firstly, Section 5.2 presents a construction that enforces the same $\mathsf{S}_1 = \cdots = \mathsf{S}_n = \mathsf{S}$ for all clients, by hashing it using a full-domain hash function modeled as a random oracle (RO), along with the tag at the time of encryption. Indeed, we will use an argument resembling what we do in the single-client construction and perform a masking procedure key by key, where the functional key query for $(\mathbb{A}, \mathbf{y}^{(\ell)})$ is indexed by ℓ. For each $i \in [n]$,we mask $(\mathbf{k}^*_{i,j})_j = (..., a^{(\ell)}_{i,j}, a'^{(\ell)}_{i,j} y/z_j, ...)_j$,$\mathbf{k}^*_{i,\mathsf{root}} = (..., a^{(\ell)}_{i,0}, a'^{(\ell)}_{i,0} y, ...)$ and $(\mathbf{c}_{i,j})_j = (..., \psi_i, \tau x z_j, ...)_j, \mathbf{c}_{i,\mathsf{root}} = (..., \psi_i, \tau x, ...)$, where $(a^{(\ell)}_{i,j})_j, (a'^{(\ell)}_{i,j})_j$ are secret shares of $a^{(\ell)}_{i,0}, a'^{(\ell)}_{i,0}$ respectively. In this more restrictive case of Section 5.2 where all n clients use the same S, it entails all clients $i \in [n]$ using the same $a^{(\ell)}_0, a'^{(\ell)}_0$ with their secret shares $(a^{(\ell)}_j)_j, (a'^{(\ell)}_j)_j$ in $(\mathbf{k}^*_{i,j})_j = (..., a^{(\ell)}_j, a'^{(\ell)}_j y/z_j, ...)_j$ and $\mathbf{k}^*_{i,\mathsf{root}} = (..., a^{(\ell)}_0, a'^{(\ell)}_0 y, ...)$. Afterwards, we want to replace $a'^{(\ell)}_0$ by an independent and uniformly random value $r^{(\ell)}_0 \xleftarrow{\$} \mathbb{Z}^*_q$ if $\mathbb{A}(\mathsf{S}_i) = 0$ and clearing the masks otherwise. As our first observation, the reasoning is still based crucially on the fact that in S there will lack some j whose corresponding z_j permits recovering $a'^{(\ell)}_0 = \sum_j z_j (a'^{(\ell)}_j / z_j)$ if $\mathbb{A}(\mathsf{S}) = 0$. It gets clear that as long as $\mathbb{A}(\mathsf{S}) = 0$, for all i independently, the same argument will hold because all i use the same set S of attributes. This observation leads to a *compression* of all $(\mathbf{c}_{i,j})_j, (\mathbf{k}^*_{i,j})_j$ into one pair of dual bases $(\mathbf{F}, \mathbf{F}^*)$ instead of n separate pairs for each $i \in [n]$. As a second observation,when $\mathbb{A}(\mathsf{S}) = 1$, all ciphertext components must be combined together for a correct decryption. As a result, to program the canceling mechanism, instead of naively embedding n pairs of constants $(\Delta\mathbf{x}[k], \mathbf{y}^{(\ell)}[k])_{k=1}^n$ in $(\mathbf{c}_{i,\mathsf{root}}, (\mathbf{c}_{i,j})_j, \mathbf{k}^*_{i,\mathsf{root}}, (\mathbf{k}^*_{i,j})_j)$ for *each* i, we only need to embed $(\Delta\mathbf{x}[i], \mathbf{y}^{(\ell)}[i])$ in $(\mathbf{c}_{i,\mathsf{root}}, (\mathbf{c}_{i,j})_j, \mathbf{k}^*_{i,\mathsf{root}}, (\mathbf{k}^*_{i,j})_j)$. The grouping by i of the products $\mathbf{c}_{i,\mathsf{root}} \times \mathbf{k}^*_{i,\mathsf{root}}$ as well as $\sum_j \mathbf{c}_{i,j} \times \mathbf{k}^*_{i,j}$ will retrieve $[\![\tau r^{(\ell)}_0 \langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle]\!]_\mathsf{t}$ and we proceed the remaining as in the single-client proof. We point out that in the multi-client setting, it might be the case that some i are corrupted and the retrieval of $[\![\tau r^{(\ell)}_0 \langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle]\!]_\mathsf{t}$ is more complicated when regrouping over i. However, by carefully defining (see

Definition 9) and considering only *admissible* adversaries, i.e. they cannot win by trivial attacks[6], it remains the case. This individual insertion of $(\Delta\mathbf{x}[i], \mathbf{y}^{(\ell)}[i])$ for each i leads to a *duplication* of one pair of dual bases $(\mathbf{H}_i, \mathbf{H}_i^*)$ for each $(\mathbf{c}_{i,\mathsf{root}}, \mathbf{k}^*_{i,\mathsf{root}})$, while all $(\mathbf{c}_{i,j})_j, (\mathbf{k}^*_{i,j})_j$ are readily put in the same basis following our first observation:

(Compressing) for all $i \in [n]$	$\mathbf{c}_{i,j}$	(	$\cdots$	ψ	$\tau\Delta\mathbf{x}[i]z_j$	$\cdots$	$)_{\mathbf{F}}$
	$\mathbf{k}^*_{i,j}$	(	$\cdots$	$a_j^{(\ell)}$	$a_j'^{(\ell)}\mathbf{y}^{(\ell)}[i]/z_j$	$\cdots$	$)_{\mathbf{F}^*}$
(Duplicating) for each $i \in [n]$	$\mathbf{c}_{i,\mathsf{root}}$	(	$\cdots$	ψ	$\tau\Delta\mathbf{x}[i]$	$\cdots$	$)_{\mathbf{H}_i}$
	$\mathbf{k}^*_{i,\mathsf{root}}$	(	$\cdots$	$a_0^{(\ell)}$	$a_0'^{(\ell)}\mathbf{y}^{(\ell)}[i]$	$\cdots$	$)_{\mathbf{H}_i^*}$

We emphasize that this parallel process is feasible thanks to a conveniently smooth control, as low as the level of the vectors' coordinates in DPVSes. This potential of parallelization helps us spread the necessary information for answering adaptive key queries, which accounts for the linearly large dimension, into n collections $\{(\mathbf{k}^*_{i,j})_{j\in\mathsf{List\text{-}Att}(\mathbb{A})}, \mathbf{k}^*_{i,\mathsf{root}}\}_{i\in[n]}$. On the one hand, we change the vectors $(\mathbf{k}^*_{i,j}, \mathbf{c}_{i,j})_{i,j}$ in parallel for all i, while these vectors are written in bases $(\mathbf{F}, \mathbf{F}^*)$. On the other hand, we change the vectors $(\mathbf{k}^*_{i,\mathsf{root}}, \mathbf{c}_{i,\mathsf{root}})_i$ independently for each client i, using the fact that each pair $(\mathbf{k}^*_{i,\mathsf{root}}, \mathbf{c}_{i,\mathsf{root}})$ belong to a separate pair of dual bases $(\mathbf{H}_i, \mathbf{H}_i^*)$. In the end, instead of using n bases of dimension n, we can use n bases of *constant* dimension for $(\mathbf{k}^*_{i,\mathsf{root}})_i$ along with one *constant*-dimension basis for all $\{(\mathbf{k}^*_{i,j})_{j\in\mathsf{List\text{-}Att}(\mathbb{A})}\}_i$, saving a factor n in the ciphertext's size.

The more flexible MCFE. Section 5.4 discusses an extension of the above MCFE construction where we do not impose the same set of attributes among n clients. Each client i can now encrypt using a different S_i and the decryption can decrypt the inner-product if and only if $\mathbb{A}(\mathsf{S}_i) = 1$ for all i. Unsurprisingly, our argument as it is from the previous construction, for masking and for replacing $a_{i,0}'^{(\ell)}$ by an independent and uniformly random value, does not hold anymore because there might be two keys corresponding to $\mathbb{A}^{(\ell)}$ and $\mathbb{A}^{(\ell')}$ such that $\mathbb{A}^{(\ell)}(\mathsf{S}_i) \neq \mathbb{A}^{(\ell')}(\mathsf{S}_i)$ and the adversary might try to use key components of the ℓ'-th query to recover $a_{i,0}'^{(\ell)}$ in the ℓ-th query. We thus make use of another layer of random secret shares $(d_{\ell,i})_{i=1}^n$ over n components of each ℓ-th functional key, facing θ_i in the ciphertext components such that $\sum_{i=1}^n \theta_i d_{\ell,i} = 0$. The values $(\theta_i)_i$ are generated as part of the master secret key but $(d_{\ell,i})_{i=1}^n$ are chosen independently for each key. A fully working key can be obtain only if all the n components corresponding to $(d_{\ell,i})_{i=1}^n$ are combined. That will prevent the adversary from trying to mix components between two different keys, i.e. if $\mathbb{A}^{(\ell)}(\mathsf{S}_i) = 0$ we can be sure that $a_{i,0}'^{(\ell)}$ retains its entropy and stays hidden. After a similar masking step using the secret shares $(a_{i,j}'^{(\ell)})_j$ of $a_{i,0}'^{(\ell)}$ independently generated for each i, the randomness provided by $(d_{\ell,i})_{i=1}^n$ allows us to tweak $a_{i,0}'^{(\ell)}$ with a uniformly random value $r_0^{(\ell)}$:

[6] For instance, the adversary might corrupt i^*, query a left-or-right challenge $(\mathbf{x}_0, \mathbf{x}_1)$ where $\Delta\mathbf{x}[i^*] := \mathbf{x}_0[i^*] - \mathbf{x}_1[i^*] \neq 0$ and $\Delta\mathbf{x}[i] = 0$ for $i \neq i^*$, then decrypt the challenge ciphertext with a satisfied key for $\mathbf{y}^{(\ell)}$ whose i^*-th entry is non-zero.

(Compressing) for all $i \in [n]$	$\mathbf{c}_{i,j}$	$(\ \cdots$	ψ	$\tau\Delta\mathbf{x}[i]z_j$	$\cdots$	$\cdots)_{\mathbf{F}}$
	$\mathbf{k}^*_{i,j}$	$(\ \cdots$	$a_j^{(\ell)}$	$a_{i,j}^{\prime(\ell)}\mathbf{y}^{(\ell)}[i]/z_j$	$\cdots$	$\cdots)_{\mathbf{F}^*}$
(Duplicating) for each $i \in [n]$	$\mathbf{c}_{i,\mathsf{root}}$	$(\ \cdots$	ψ	$\tau\Delta\mathbf{x}[i]$	θ_i	$\cdots)_{\mathbf{H}_i}$
	$\mathbf{k}^*_{i,\mathsf{root}}$	$(\ \cdots$	$a_0^{(\ell)}$	$(a_{i,0}^{\prime(\ell)} + r_0^{(\ell)})\mathbf{y}^{(\ell)}[i]$	$d_{\ell,i}$	$\cdots)_{\mathbf{H}_i^*}$

It is of the utmost importance that we rely on $(d_{\ell,i})_{i=1}^n$, which is particular for each ℓ-th key, to carry out this change from $a_{i,0}^{\prime(\ell)}$ to $a_{i,0}^{\prime(\ell)} + r_0^{(\ell)}$. Or else, the adversary can mix and match the ℓ-th and ℓ'-th keys to remove $a_{i,0}^{\prime(\ell)}$ and distinguish the adding of $r_0^{(\ell)}$, regardless whether S_i is authorized or not. The argument is now computational, in contrast to the information-theoretical indistinguishability when changing from $a_0^{\prime(\ell)}$ to $r_0^{(\ell)}$ in the more restrictive MCFE. We now perform an unmasking by going backwards to remove the sharing $(a_{i,j}^{\prime(\ell)})_j$ and $a_{i,0}^{\prime(\ell)}$ in the key. This transition is completely symmetric. If $\mathbb{A}(\mathsf{S}_i) = 1$ for all i, then the admissibility requires $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle = 0$ and the noise $\tau r_0^{(\ell)}$ can be removed. Otherwise, in case $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle \neq 0$, the mask $\tau r_0^{(\ell)}$ persists but the admissbility implies there exists i such that $\mathbb{A}(\mathsf{S}_i) = 0$ and the functional key cannot decrypt the challenge ciphertext. We emphasize that the incapability of the key when $\mathbb{A}(\mathsf{S}_i) = 0$ is ensured by $(d_{\ell,i})_{i=1}^n$. After introducing $r_0^{(\ell)}$, the remaining steps resemble the proof of the less flexible construction in Section 5.2. A desirable byproduct of this more flexible construction is that the hash function, which is modeled as a random oracle (RO), is now applied only on the tag. Therefore, we can obtain an MIFE in the standard model that is comparable to the work in [4] by fixing the hash value of a tag for all ciphertexts and publishing it as a parameter of the scheme.

4 Single-Client Functional Encryption For Inner-product with Fine-Grained Access Control via LSSS

We present constructions of FE for the inner-product functionality with attribute-based control expressed using linear secret sharing schemes, starting with the simpler single-client setting. We are in the bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ and $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}$ are written additively. The function class of interests is $\mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$ where $\mathcal{F}^{\mathsf{IP}}$ contains $F_\mathbf{y} : \left(\mathbb{Z}_q^*\right)^n \to \mathbb{Z}_q$ defined as $F_\mathbf{y}(\mathbf{x}) := \langle \mathbf{x}, \mathbf{y} \rangle$. The access control is given by $\mathsf{Rel} : \mathsf{LSSS} \times 2^{\mathsf{Att}} \to \{0, 1\}$, where $\mathsf{Rel}(\mathbb{A}, \mathsf{S}) = \mathbb{A}(\mathsf{S})$, the class LSSS contains Linear Secret Sharing Schemes over Att, and 2^{Att} denotes the superset of an attribute space $\mathsf{Att} \subseteq \mathbb{Z}_q$. Our constructions are key-policy, where $\mathbb{A}$ is embedded in the key and S is specified in the ciphertext. In order to facilitate the understanding and the motivation of our later multi-client constructions in Section 5, we present both selectively-secure and adaptively-secure single-client constructions in Figure 1. We leverage the selectively-secure scheme to obtain the adaptively-secure one by replacing certain elements in the former by the corresponding boxed components for the latter.

The main difference between the adaptive version and the selectively-secure version is the increase in the dimension of dual bases, from constant dimensions

$\underline{\mathsf{Setup}(1^\lambda)}$**:** Choose two pairs of dual orthogonal bases $(\mathbf{F}, \mathbf{F}^*)$ and $(\mathbf{H}, \mathbf{H}^*)$ where $(\mathbf{H}, \mathbf{H}^*)$ is a pair of bases of the dual pairing vector spaces $(\mathbb{G}_1^4, \mathbb{G}_2^4)$ $\boxed{(\mathbb{G}_1^{n+3}, \mathbb{G}_2^{n+3})}$, and $(\mathbf{F}, \mathbf{F}^*)$ are dual bases of $(\mathbb{G}_1^8, \mathbb{G}_2^8)$ $\boxed{(\mathbb{G}_1^{n+7}, \mathbb{G}_2^{n+7})}$. We write

$$\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3, \mathbf{h}_4) \qquad \mathbf{H}^* = (\mathbf{h}_1^*, \mathbf{h}_2^*, \mathbf{h}_3^*, \mathbf{h}_4^*)$$
$$\boxed{\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3, \mathbf{h}_4, \ldots, \mathbf{h}_{n+3})} \qquad \boxed{\mathbf{H}^* = (\mathbf{h}_1^*, \mathbf{h}_2^*, \mathbf{h}_3^*, \mathbf{h}_4^*, \ldots, \mathbf{h}_{n+3}^*)}$$
$$\mathbf{F} = (\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3, \mathbf{f}_4, \mathbf{f}_5, \mathbf{f}_6, \mathbf{f}_7, \mathbf{f}_8) \qquad \mathbf{F}^* = (\mathbf{f}_1^*, \mathbf{f}_2^*, \mathbf{f}_3^*, \mathbf{f}_4^*, \mathbf{f}_5^*, \mathbf{f}_6^*, \mathbf{f}_7^*, \mathbf{f}_8^*)$$
$$\boxed{\mathbf{F} = (\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3, \mathbf{f}_4, \ldots, \mathbf{f}_{n+5}, \mathbf{f}_{n+6}, \mathbf{f}_{n+7})} \qquad \boxed{\mathbf{F}^* = (\mathbf{f}_1^*, \mathbf{f}_2^*, \mathbf{f}_3^*, \mathbf{f}_4^*, \ldots, \mathbf{f}_{n+5}^*, \mathbf{f}_{n+6}^*, \mathbf{f}_{n+7}^*)}$$

and sample $\mu, z \xleftarrow{\$} \mathbb{Z}_q^*, S, U \xleftarrow{\$} (\mathbb{Z}_q^*)^n$ and write $S = (s_1, \ldots, s_n)$, $U = (u_1, \ldots, u_n)$. Output the public key and the master secret key as

$$\begin{cases} \mathsf{pk} := \left(\mathbf{h}_1 + \mu\mathbf{h}_2,\ \mathbf{h}_3,\ (\mathbf{f}_i)_{i\in[3]},\ (\llbracket s_i + \mu \cdot u_i \rrbracket_1)_{i\in[n]}\right) \\ \mathsf{msk} := (z,\ S,\ U,\ (\mathbf{f}_i^*)_{i\in[3]},\ (\mathbf{h}_i^*)_{i\in[3]})\ . \end{cases}$$

$\underline{\mathsf{Extract}(\mathsf{msk}, \mathbb{A}, \mathbf{y} \in \mathbb{Z}_q^n)}$**:** Let $\mathbb{A}$ be an LSSS-realizable monotone access structure over a set of attributes $\mathsf{Att} \subseteq \mathbb{Z}_q$. First, sample $a_0 \xleftarrow{\$} \mathbb{Z}_q$ and run the labeling algorithm $\Lambda_{a_0}(\mathbb{A})$ (see (1)) to obtain the labels $(a_j)_j$ where j runs over the attributes in Att. In the end, it holds that $a_0 = \sum_{j\in A} c_j \cdot a_j$ where j runs over an authorized set $A \in \mathbb{A}$ and $\mathbf{c}_A = (c_j)_{j\in A}$ is the reconstruction vector from LSSS w.r.t A. We denote by $\mathsf{List\text{-}Att}(\mathbb{A})$ the list of attributes appearing in $\mathbb{A}$, with possible repetitions. Parse $\mathsf{msk} = (z,\ S,\ U,\ (\mathbf{f}_i^*)_{i\in[3]},\ (\mathbf{h}_i^*)_{i\in[3]})$. Compute:

$$\mathbf{k}_j^* := (\pi_j \cdot (j, 1),\ a_j \cdot z,\ 0,\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}^*} \text{ for } j \in \mathsf{List\text{-}Att}(\mathbb{A})$$
$$\boxed{\mathbf{k}_j^* := (\pi_j \cdot (j, 1),\ a_j \cdot z, \overbrace{0, \ldots, 0}^{n \text{ times}},\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}^*} \text{ for } j \in \mathsf{List\text{-}Att}(\mathbb{A})}$$
$$\mathbf{m}_i^* := \llbracket \mathbf{y}[i] \rrbracket_2 \text{ for } i \in [n]$$
$$\mathbf{k}_{\mathsf{ipfe}}^* := (\langle S, \mathbf{y}\rangle, \langle U, \mathbf{y}\rangle,\ a_0 \cdot z,\ 0)_{\mathbf{H}^*} \qquad \boxed{\mathbf{k}_{\mathsf{ipfe}}^* := (\langle S, \mathbf{y}\rangle, \langle U, \mathbf{y}\rangle,\ a_0 \cdot z, \overbrace{0, \ldots, 0}^{n \text{ times}})_{\mathbf{H}^*}}$$

where $\pi_j \xleftarrow{\$} \mathbb{Z}_q$. Output $\mathsf{sk}_{\mathbb{A},\mathbf{y}} := \left(\left(\mathbf{k}_j^*\right)_j, (\mathbf{m}_i^*)_{i\in[n]}, \mathbf{k}_{\mathsf{ipfe}}^*\right)$.

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathbf{x}, \mathsf{S})}$**:** Parse the public key $\mathsf{pk} = \left(\mathbf{h}_1 + \mu\mathbf{h}_2,\ \mathbf{h}_3,\ (\mathbf{f}_i)_{i\in[3]},\ (\llbracket s_i + \mu \cdot u_i \rrbracket_1)_{i\in[n]}\right)$ and $\mathsf{S} \subseteq \mathsf{Att} \subseteq \mathbb{Z}_q$ as the set of attributes, then sample $\omega, \psi \xleftarrow{\$} \mathbb{Z}_q$. Compute

$$\mathbf{c}_j = \sigma_j \cdot \mathbf{f}_1 - j \cdot \sigma_j \cdot \mathbf{f}_2 + \psi \cdot \mathbf{f}_3 = (\sigma_j \cdot (1, -j),\ \psi,\ 0,\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}} \text{ for each } j \in \mathsf{S}$$
$$\boxed{\mathbf{c}_j = (\sigma_j \cdot (1, -j),\ \psi, \overbrace{0, \ldots, 0}^{n \text{ times}}, 0,\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}} \text{ for each } j \in \mathsf{S}}$$

where $\sigma_j \xleftarrow{\$} \mathbb{Z}_q$. Finally, compute

$$\mathbf{t}_i = \omega \cdot \llbracket s_i + \mu \cdot u_i \rrbracket_1 + \llbracket \mathbf{x}[i] \rrbracket_1 = \llbracket \omega \cdot (s_i + \mu u_i) + \mathbf{x}[i] \rrbracket_1 \text{ for } i \in [n]$$
$$\mathbf{c}_{\mathsf{ipfe}} = \omega \cdot (\mathbf{h}_1 + \mu\mathbf{h}_2) + \psi \cdot \mathbf{h}_3 = (\omega,\ \mu\omega,\ \psi,\ 0)_{\mathbf{H}} \qquad \boxed{\mathbf{c}_{\mathsf{ipfe}} = (\omega,\ \mu\omega,\ \psi,\ \overbrace{0, \ldots, 0}^{n \text{ times}})_{\mathbf{H}}}$$

where $\sigma_i \xleftarrow{\$} \mathbb{Z}_q$ for every $i \in [n]$ and output $\mathsf{ct} := \left((\mathbf{c}_j)_{j\in\mathsf{S}}, (\mathbf{t}_i)_{i\in[n]}, \mathbf{c}_{\mathsf{ipfe}}\right)$.

$\underline{\mathsf{Dec}(\mathsf{sk}_{\mathbb{A},\mathbf{y}}, \mathsf{ct})}$**:** Parse $\mathsf{ct} = \left((\mathbf{c}_j)_{j\in\mathsf{S}},\ (\mathbf{t}_i)_{i\in[n]},\ \mathbf{c}_{\mathsf{ipfe}}\right)$ and $\mathsf{sk}_{\mathbb{A},\mathbf{y}} := \left(\left(\mathbf{k}_j^*\right)_{j\in\mathsf{List\text{-}Att}(\mathbb{A})},\ (\mathbf{m}_i^*)_{i\in[n]},\ \mathbf{k}_{\mathsf{ipfe}}^*\right)$. If there exists $A \subseteq \mathsf{S}$ and $A \in \mathbb{A}$, then compute the reconstruction vector $\mathbf{c} = (c_j)_j$ of the LSSS for A and

$$\llbracket \mathsf{out} \rrbracket_{\mathsf{t}} = \sum_{j\in A} \mathbf{c}_j \times (c_j \cdot \mathbf{k}_j^*) + \sum_{i=1}^{n} (\mathbf{e}(\mathbf{t}_i, \mathbf{m}_i^*)) - \left(\mathbf{c}_{\mathsf{ipfe}} \times \mathbf{k}_{\mathsf{ipfe}}^*\right)$$

Finally, compute the discrete logarithm and output $\mathsf{out} \in \mathbb{Z}_q$. Else, output $\perp$.

Fig. 1: The selectively-secure and $\boxed{\text{adaptively-secure}}$ single-client constructions for IPFE with fine-grained access control via LSSS. The high-level ideas can be found in the technical overview of Section 3 and more details are presented in the full version [34].

to dimensions linear in n. The details can be found in Figure 1. The computation for encrypting and decrypting stays essentially the same. We refer to the technical overview in Section 3 for the main ideas why using bigger DPVSes allows us to achieve the stronger adaptive notion. The *correctness* can be verified in a straightforward manner. Theorem 7 proves the adaptive IND-security for the construction corresponding to boxed components in Figure 1, where the adversary can query a unique challenge ciphertext and multiple functional keys. Using a standard hybrid argument and recalling that our scheme is public-key provide us with adaptive security against multiple challenge ciphertexts. The easier selective security can be proved using similar techniques. Full details can be found in the full version [34].

Theorem 7. *Let* $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ *be an IPFE scheme with fine-grained access control via LSSS presented in Figure 1 in a bilinear group setting* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$*, for the functionality class* $\mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$*. Then,* $\mathcal{E}$ *is secure against chosen-plaintext attacks, adaptively in the attributes and the challenge messages, if the* SXDH *assumption holds for* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*. More precisely, for* $\lambda \in \mathbb{N}$ *and for any* ppt *adversary* $\mathcal{A}$*, let* n *be the dimension of vectors for inner-product computation,* K *denote the total number of functional key queries, and* P *denote the total number of attributes used by the adversary. We have the following bound:*

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{E},\mathcal{F}^{\mathsf{IP}},\mathsf{LSSS},\mathcal{A}}(1^\lambda) \leq (2nK \cdot (P(6P+3)+2)+5) \cdot \mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda)$$

where $\mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda)$ *denotes the maximum advantage over* ppt *adversaries against the* SXDH *problem in* $(\mathbb{G}_1, \mathbb{G}_2)$ *set up with parameter* λ*.*

5 Multi-Client Functional Encryption for Inner-Product with Fine-Grained Access Control via LSSS

First of all, we define and give the model of security for *multi-client functional encryption with fine-grained access control* in Section 5.1. We then present our main contribution by extending our FE scheme in Section 4 from the single-client setting to the multi-client setting in Section 5.2, for the functionality class to evaluate inner-products under access control by linear secret-sharing schemes. Theorem 14 proves its adaptive security. Finally, in Section 5.4 we discuss further our construction and revisit the MIFE regime for comparison with [4].

5.1 Definitions

We extend the notion of functional encryption with fine-grained access control to the multi-client setting. The access control is defined via a relation $\mathsf{Rel} : \mathsf{AC\text{-}K} \times \mathsf{AC\text{-}Ct}_1 \times \cdots \times \mathsf{AC\text{-}Ct}_n \to \{0,1\}$, for some sets $\mathsf{AC\text{-}Ct}_1, \ldots, \mathsf{AC\text{-}Ct}_n$ and $\mathsf{AC\text{-}K}$. A plaintext for client i consists of $(\mathsf{ac\text{-}ct}_i, x_i) \in \mathsf{AC\text{-}Ct}_i \times \mathcal{D}_\lambda$, whose corresponding ciphertext can be decrypted to $F_\lambda(x)$ using the functional key $\mathsf{sk}_{F_\lambda,\mathsf{ac\text{-}k}}$ for $\mathsf{ac\text{-}k} \in \mathsf{AC\text{-}K}$ if and only if $\mathsf{Rel}(\mathsf{ac\text{-}k}, (\mathsf{ac\text{-}ct}_i)_i) = 1$.

Definition 8 (Multi-client functional encryption with fine-grained access control). *A* multi-client functional encryption (MCFE) scheme with fine-grained access control *for the functionality class* $\mathcal{F} \times \mathsf{AC\text{-}K}$ *consists of four algorithms* $(\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$*:*

$\mathsf{Setup}(1^\lambda)$**:** *Given as input a security parameter* λ*, output a master secret key* msk *and* $n = n(\lambda)$ *encryption keys* $(\mathsf{ek}_i)_{i\in[n]}$ *where* $n : \mathbb{N} \to \mathbb{N}$ *is a function.*

$\mathsf{Extract}(\mathsf{msk}, F_\lambda, \mathsf{ac\text{-}k})$**:** *Given* $\mathsf{ac\text{-}k} \in \mathsf{AC\text{-}K}$*, a function description* $F_\lambda \in \mathcal{F}$*, and the master secret key* msk*, output a decryption key* $\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}}$*.*

$\mathsf{Enc}(\mathsf{ek}_i, x_i, \mathsf{tag}, \mathsf{ac\text{-}ct}_i)$**:** *Given as inputs* $\mathsf{ac\text{-}ct}_i \in \mathsf{AC\text{-}Ct}_i$*, an encryption key* ek_i*, a message* $x_i \in \mathcal{D}_\lambda$*, and a tag* tag*, output a ciphertext* $\mathsf{ct}_{\mathsf{tag},i}$*.*

$\mathsf{Dec}(\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}}, \mathbf{c})$**:** *Given the decryption key* $\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}}$ *and a vector of ciphertexts* $\mathbf{c} := (\mathsf{ct}_{\mathsf{tag},i})_i$ *of length* n*, output an element in* $\mathcal{R}_\lambda$ *or an invalid symbol* $\perp$*.*

Correctness. For sufficiently large $\lambda \in \mathbb{N}$, for all $(\mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda)$, $(F_\lambda, \mathsf{ac\text{-}k}) \in \mathcal{F} \times \mathsf{AC\text{-}K}$ and $\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}} \leftarrow \mathsf{Extract}(\mathsf{msk}, F_\lambda, \mathsf{ac\text{-}k})$, for all tag and $(\mathsf{ac\text{-}ct}_i)_i$ satisfying $\mathsf{Rel}(\mathsf{ac\text{-}k}, (\mathsf{ac\text{-}ct}_i)_i) = 1$, for all $(x_i)_{i\in[n]} \in \mathcal{D}_\lambda^n$, if $F_\lambda(x_1, \ldots, x_n) \neq \perp$, the following holds with overwhelming probability:

$$\mathsf{Dec}\left(\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}}, (\mathsf{Enc}(\mathsf{ek}_i, x_i, \mathsf{tag}, \mathsf{ac\text{-}ct}_i))_{i\in[n]}\right) = F_\lambda(x_1, \ldots, x_n)$$

where $F_\lambda : \mathcal{D}_\lambda^n \to \mathcal{R}_\lambda$ and the probability is taken over the coins of algorithm.

Security. We define an indistinguishability-based security notion taking into account the attribute-based access control as well as the possibility of collusion among multiple clients. Below we define the *admissibility* of an adversary $\mathcal{A}$ in the security game against $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$. Intuitively, we consider only admissible adversaries who do not win our security game in a trivial manner as well as other meaningful restrictions in the multi-client setting. The admissibility additionally takes into account the satisfiability of the relation for access control, which also complicates the way we model the security notion. In the plain setting, interested readers can refer to [20] or [32] for more details.

Definition 9 (Admissible adversaries). *Let* $\mathcal{A}$ *be a* ppt *adversary and let* $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ *be an MCFE scheme with fine-grained access control for the functionality class* $\mathcal{F} \times \mathsf{AC\text{-}K}$*. In the security game given in Figure 2 for* $\mathcal{A}$ *considering* $\mathcal{E}$*, let the sets* $(\mathcal{C}, \mathcal{Q}, \mathcal{H})$ *be the sets of corrupted clients, functional key queries, and honest clients, in that order. We say that* $\mathcal{A}$ is NOT admissible w.r.t $(\mathcal{C}, \mathcal{Q}, \mathcal{H})$ *if any of the following conditions holds:*

1. *There exist two different partial ciphertexts for* $x_i^{(b)} \neq x_i^{(b)\prime}$*, for some* $b \in \{0, 1\}$*, under one challenge tag* tag *that is queried to* $\mathbf{LoR}$*.*
2. *There exist a tag* tag *and* $i, j \in \mathcal{H}$ *such that* $i \neq j$*, there exists a query* $(i, x_i^{(0)}, x_i^{(1)}, \mathsf{tag}, \mathsf{ac\text{-}ct}_i)$ *to* $\mathbf{LoR}$ *but there exist no query* $(j, x_j^{(0)}, x_j^{(1)}, \mathsf{tag}, \mathsf{ac\text{-}ct}_j)$ *to* $\mathbf{LoR}$*.*
3. *There exists* $(\mathsf{tag}, \mathsf{ac\text{-}ct}_i)$ *for* $i \in [n]$*, a function* $F \in \mathcal{F}$*, and* $\mathsf{ac\text{-}k} \in \mathsf{AC\text{-}K}$ *such that*
   - *We have* $\mathsf{Rel}(\mathsf{ac\text{-}k}, (\mathsf{ac\text{-}ct}_i)_i) = 1$ *and* $(F, \mathsf{ac\text{-}k}) \in \mathcal{Q}$*.*

- *For all* $i \in \mathcal{H}$*, there exists a query* $(i, x_i^{(0)}, x_i^{(1)}, \mathsf{tag}, \mathsf{ac\text{-}ct}_i)$ *to* **LoR** *for* $(x_i^{(0)}, x_i^{(1)})$.
- *For all* $i \in \mathcal{C}$*, it holds that* $x_i^{(0)} = x_i^{(1)}$.

Otherwise, we say that $\mathcal{A}$ is admissible w.r.t $(\mathcal{C}, \mathcal{Q}, \mathcal{H})$.

Remark 10. As in the plain MCFE with no attribute-based access control in [20, 32], we will consider security with no repetitions, i.e. the adversary cannot query **Enc** nor **LoR** for multiple ciphertexts under the same $(i, \mathsf{tag}, \mathsf{ac\text{-}ct}_i)$. Moreover, the adversary is not allowed to query the encryption oracle **Enc** for ciphertexts under the challenge tag^* that was previously queried to **LoR**. The intuition of this restriction is to prevent trivial attacks where, by querying for ciphertexts under tag^*, the adversary can combine them with the challenge ciphertext under the same tag^* to learn much more information about the challenge bit b and win the game. In addition, for every honest clients i, there must be a ciphertext query to **LoR** under the challenge $(\mathsf{tag}, \mathsf{ac\text{-}ct}_i)$. That is, we do not take into account the scenario where only partial (in terms of honest clients) challenge ciphertext is queried to **LoR**. We can relax this condition and allow partial challenge ciphertexts by adding a layer of *All-or-Nothing Encapsulation* (AoNE). The AoNE encapsulates the partial components from clients and guarantees that all encapsulated components can be decapsulated if and only if all components are gathered, otherwise the original information remain hidden. The work by Chotard *et al.* [22] presents constructions for AoNE in the prime-order (asymmetric) bilinear groups compatible with our current setting. In the MIFE realm, the work of [4] considers the similar restriction and expects all honest slots $i \in [n]$ are queried to **LoR**.

Remark 11. Our syntax and model of MCFE with fine-grained access control require that in order to combine the ciphertext components, they must be encrypted under the same tag and the same set of attributes. One can aim for a more flexible notion in which each client i can encrypt their ciphertext component under a different $(\mathsf{tag}, \mathsf{ac\text{-}ct}_i)$. However, this creates a much more intricate situation and we have to take into account non-trivial attacks where two different functional keys, whose policies are satisfied by different subsets of clients, may be combined to evaluate the underlying plaintext components of the union of the foregoing subsets. By hashing the tags and attributes during encryption, our concrete constructions enforce the same set of attributes embedded in the ciphertext components. In Section 5.4, we discuss how to relax the constraint and achieve the flexible notion where each client i can use a different $(\mathsf{tag}, \mathsf{ac\text{-}ct}_i)$ and hash only tag. As a result, this more flexible MCFE scheme in the RO model can be morphed into an MIFE scheme in the *standard* model by fixing a public tag and publishing its hash.

We are now ready to give the definition for the indistinguishability-based security.

Definition 12 (IND-security for MCFE with fine-grained access control). *An MCFE scheme with fine-grained access control* $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract},$

$\mathsf{Enc}, \mathsf{Dec}$) *for the functionality class* $\mathcal{F} \times \mathsf{AC\text{-}K}$ *is* IND-secure *if for all* ppt *adversaries* $\mathcal{A}$*, and for all sufficiently large* $\lambda \in \mathbb{N}$*, the following probability is negligible*

$$\mathsf{Adv}^{\mathsf{mc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda) \coloneqq \left|\Pr[\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2}\right| \ .$$

The game $\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda)$ *is depicted in Figure 2. The probability is taken over the random coins of* $\mathcal{A}$ *and the algorithms.*

In a more relaxed notion, the scheme $\mathcal{E}$ *is* selectively IND-secure *if the following probability is negligible*

$$\mathsf{Adv}^{\mathsf{mc\text{-}sel\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda) \coloneqq \left|\Pr[\mathsf{Expr}^{\mathsf{mc\text{-}sel\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2}\right| \ .$$

We also define a notion of security where only one challenge tag tag^* *is allowed. That is, the scheme* $\mathcal{E}$ *is* one-time IND-secure *if the following probability is negligible*

$$\mathsf{Adv}^{\mathsf{mc\text{-}ind\text{-}cpa\text{-}1\text{-}chal}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda) \coloneqq \left|\Pr[\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}cpa\text{-}1\text{-}chal}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2}\right| \ .$$

Lemma 13 allows us to concentrate on the notion of one-time IND-security for our construction. The proof is a standard hybrid argument and we give it in the full version [34] for completeness.

Lemma 13. *Let* $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ *for the function class* $\mathcal{F} \times \mathsf{AC\text{-}K}$ *be an MCFE scheme with fine-grained access control. If* $\mathcal{E}$ *is one-time IND-secure, then* $\mathcal{E}$ *is IND-secure.*

5.2 Construction

This section presents a multi-client FE scheme with fine-grained access control, as defined in Section 5.1. We are in the bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ and $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}$ are written additively. In our concrete construction, the functionality class of interests is $\mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$ and $\mathcal{F}^{\mathsf{IP}}$ contains $F_\mathbf{y} : \left(\mathbb{Z}_q^*\right)^n \to \mathbb{Z}_q$ that is defined as $F_\mathbf{y}(\mathbf{x}) \coloneqq \langle \mathbf{x}, \mathbf{y} \rangle$. The access control is given by $\mathsf{Rel} : \mathsf{LSSS} \times \left(\prod_{i=1}^n 2^{\mathsf{Att}}\right) \to \{0,1\}$, where $\mathsf{Rel}(\mathbb{A}, (\mathsf{S}_i)_i) = \prod_i \mathbb{A}(\mathsf{S}_i)$, the class LSSS contains Linear Secret Sharing Schemes over Att, and 2^{Att} denotes the superset of an attribute space $\mathsf{Att} \subseteq \mathbb{Z}_q$. Our constructions are key-policy, where $\mathbb{A}$ is embedded in the key and S is specified in the ciphertext. The tag space Tag contains the tags that accompany plaintext components at the time of encryption.

We also need a full domain hash function $\mathsf{H} : \mathsf{Tag} \times 2^{\mathsf{Att}} \to \mathbb{G}_1^2$, where Tag denotes the set of tags and 2^{Att} contains the subsets of attributes of Att. The details of our construction is given in Figure 3. We remark that currently all clients $i \in [n]$ must use the same S for encrypting their inputs x_i, because S is hashed together with tag by H. Section 5.4 presents another construction that relaxes

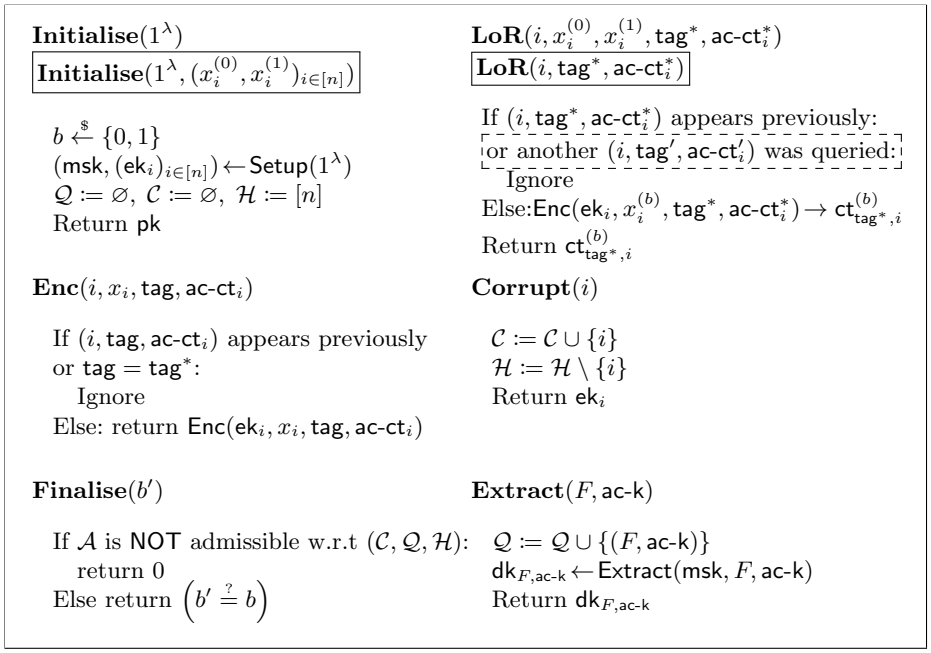

Fig. 2: The security games $\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda)$, $\boxed{\mathsf{Expr}^{\mathsf{mc\text{-}sel\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda)}$ and $\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}cpa\text{-}1\text{-}chal}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda)$ for Definition 12

the matching condition on S and H then receives only tag as inputs. We note that the *duplicate-and-compress* technique is used by putting the vectors $\{(\mathbf{c}_{i,j}, \mathbf{k}_{i,j})_j\}$ in the same pair of dual bases $(\mathbf{F}, \mathbf{F}^*)$ for all client $i \in [n]$, meanwhile each pair of vectors $(\mathbf{c}_{i,\mathsf{ipfe}}, \mathbf{k}_{i,\mathsf{ipfe}})$ is put in bases $(\mathbf{H}_i, \mathbf{H}_i^*)$ for each client $i \in [n]$. In the proof of Theorem 14 we detail how the basis changes in Lemma 4 can be done in parallel for $(\mathbf{H}_i, \mathbf{H}_i^*), (\mathbf{F}, \mathbf{F}^*)$ for all $i \in [n]$. The *correctness* of the scheme is verified by:

$$\begin{aligned}
[\![\mathsf{out}]\!]_\mathsf{t} &= \sum_{i=1}^n \left(\left(\sum_{j\in A} \mathbf{c}_{i,j} \times (c_j \cdot \mathbf{k}_{i,j}) \right) - (\mathbf{c}_{i,\mathsf{ipfe}} \times \mathbf{k}_{i,\mathsf{ipfe}}) + \mathbf{e}(\mathbf{t}_i, \mathbf{m}_i) \right) \\
&= \sum_{i=1}^n \Big([\![\psi_i a_0 z]\!]_\mathsf{t} - [\![\omega p_i \cdot \langle S, \mathbf{y}\rangle + \omega' p_i \cdot \langle U, \mathbf{y}\rangle + \psi_i a_0 z]\!]_\mathsf{t} \\
&\quad + [\![(\omega s_i + \omega' u_i + x_i) y_i]\!]_\mathsf{t} \Big) = [\![\langle \mathbf{x}, \mathbf{y}\rangle]\!]_\mathsf{t}
\end{aligned}$$

5.3 Adaptive Security

We now present the main ideas of the adaptive proof for the multi-client construction described in Section 5.2, the detailed proof is presented in the full version [34]. A high-level intuition can be revisited in Section 3.

$\underline{\mathsf{Setup}(1^\lambda)}$**:** Choose $n+1$ pairs of dual orthogonal bases $(\mathbf{H}_i, \mathbf{H}_i^*)$ for $i \in [n]$ and $(\mathbf{F}, \mathbf{F}^*)$where $(\mathbf{H}_i, \mathbf{H}_i^*)$ is a pair of dual bases for $(\mathbb{G}_1^4, \mathbb{G}_2^4)$ and $(\mathbf{F}, \mathbf{F}^*)$ is a pair of dual bases for $(\mathbb{G}_1^8, \mathbb{G}_2^8)$. We denote the basis changing matrices for $(\mathbf{F}, \mathbf{F}^*), (\mathbf{H}, \mathbf{H}_i^*)$ as $(F, F' \coloneqq (F^{-1})^\top), (H_i, H_i' \coloneqq (H_i^{-1})^\top)$ respectively (see the full version [34] for basis changes in DPVS):

$$(\mathbf{H}_i = H_i \cdot \mathbf{T};\ \mathbf{H}_i^* = H_i' \cdot \mathbf{T}^*)_{i\in[n]} \quad (\mathbf{F} = F \cdot \mathbf{W};\ \mathbf{F}^* = F' \cdot \mathbf{W}^*)$$

where $H_i, H_i' \in \mathbb{Z}_q^{4\times 4}, F, F' \in \mathbb{Z}_q^{8\times 8}$ and $(\mathbf{T} = [\![I_4]\!]_1, \mathbf{T}^* = [\![I_4]\!]_2)$, $(\mathbf{W} = [\![I_8]\!]_1, \mathbf{W}^* = [\![I_8]\!]_2)$ are canonical bases of $(\mathbb{G}_1^4, \mathbb{G}_2^4)$, $(\mathbb{G}_1^8, \mathbb{G}_2^8)$ respectively, for identity matrices I_4 and I_8. We recall that in the multi-client setting the scheme must be a private key encryption scheme. For each $i \in [n]$, we write

$$\mathbf{H}_i = (\mathbf{h}_{i,1}, \mathbf{h}_{i,2}, \mathbf{h}_{i,3}, \mathbf{h}_{i,4}) \qquad \mathbf{H}_i^* = (\mathbf{h}_{i,1}^*, \mathbf{h}_{i,2}^*, \mathbf{h}_{i,3}^*, \mathbf{h}_{i,4}^*)$$
$$\mathbf{F} = (\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3, \mathbf{f}_4, \mathbf{f}_5, \mathbf{f}_6, \mathbf{f}_7, \mathbf{f}_8) \qquad \mathbf{F}^* = (\mathbf{f}_1^*, \mathbf{f}_2^*, \mathbf{f}_3^*, \mathbf{f}_4^*, \mathbf{f}_5^*, \mathbf{f}_6^*, \mathbf{f}_7^*, \mathbf{f}_8^*)$$

and sample $\mu \xleftarrow{\$} \mathbb{Z}_q^*, S, U, \xleftarrow{\$} (\mathbb{Z}_q^*)^n$ and write $S = (s_1, \ldots, s_n)$, $U = (u_1, \ldots, u_n)$. Perform an n-out-of-n secret sharing on 1, that is, choose $p_i \in \mathbb{Z}_q$ such that $1 = p_1 + \cdots + p_n$. Output the master secret key and the encryption keys as

$$\begin{cases} \mathsf{msk} \coloneqq (S,\ U,\ \mathbf{f}_1^*,\ \mathbf{f}_2^*,\ \mathbf{f}_3^*,\ (\mathbf{h}_{i,1}^*, \mathbf{h}_{i,2}^*, \mathbf{h}_{i,3}^*)_{i\in[n]}) \\ \mathsf{ek}_i \coloneqq (s_i,\ u_i,\ p_i \cdot H_i^{(1)},\ p_i \cdot H_i^{(2)},\ \mathbf{h}_{i,3},\ \mathbf{f}_1,\ \mathbf{f}_2,\ \mathbf{f}_3) \text{ for } i \in [n] \end{cases}$$

where $H_i^{(k)}$ denotes the k-th row of H_i.

$\underline{\mathsf{Extract}(\mathsf{msk}, \mathbb{A}, \mathbf{y} \in \mathbb{Z}_q^n)}$**:** Let $\mathbb{A}$ be an LSSS-realizable monotone access structure over a set of attributes $\mathsf{Att} \subseteq \mathbb{Z}_q$. First, sample $a_0 \xleftarrow{\$} \mathbb{Z}_q$ and run the labeling algorithm $\Lambda_{a_0}(\mathbb{A})$ (see Definition 1) to obtain the labels $(a_j)_j$ where j runs over the attributes in Att. In the end, it holds that $a_0 = \sum_{j\in A} c_j \cdot a_j$ where j runs over an authorized set $A \in \mathbb{A}$ and $\mathbf{c} = (c_j)_j$ is the reconstruction vector from LSSS w.r.t A. We denote by $\mathsf{List\text{-}Att}(\mathbb{A})$ the list of attributes appearing in $\mathbb{A}$, with possible repetitions. Parse $\mathsf{msk} = (S,\ U,\ \mathbf{f}_1^*,\ \mathbf{f}_2^*,\ \mathbf{f}_3^*,\ (\mathbf{h}_{i,1}^*, \mathbf{h}_{i,2}^*, \mathbf{h}_{i,3}^*)_{i\in[n]})$ and write $\mathbf{y} = (y_1, \ldots, y_n)$. For each $i \in [n]$, compute $\mathbf{m}_i \coloneqq [\![y_i]\!]_2$ and

$$\mathbf{k}_{i,j} = (\pi_{i,j} \cdot (j, 1),\ a_j \cdot z,\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}^*} \text{ for } j \in \mathsf{List\text{-}Att}(\mathbb{A})$$
$$\mathbf{k}_{i,\mathsf{ipfe}} \coloneqq (\langle S, \mathbf{y}\rangle, \langle U, \mathbf{y}\rangle,\ a_0 \cdot z,\ 0)_{\mathbf{H}_i^*}$$

where $z, \pi_{i,j} \xleftarrow{\$} \mathbb{Z}_q$. Output $\mathsf{dk}_{\mathbb{A},\mathbf{y}} \coloneqq \left((\mathbf{k}_{i,j})_{i,j}, (\mathbf{m}_i, \mathbf{k}_{i,\mathsf{ipfe}})_{i\in[n]}\right)$.

$\underline{\mathsf{Enc}(\mathsf{ek}_i, x_i, \mathsf{tag}, \mathsf{S})}$**:** Parse $\mathsf{ek}_i \coloneqq (s_i,\ u_i,\ p_i \cdot H_i^{(1)},\ p_i \cdot H_i^{(2)},\ \mathbf{h}_{i,3},\ \mathbf{f}_1,\ \mathbf{f}_2,\ \mathbf{f}_3)$ and $\mathsf{S} \subseteq \mathsf{Att} \subseteq \mathbb{Z}_q$ as the set of attributes, compute $\mathsf{H}(\mathsf{tag}, \mathsf{S}) \to ([\![\omega]\!]_1, [\![\omega']\!]_1) \in \mathbb{G}_1^2$ and sample $\psi_i \xleftarrow{\$} \mathbb{Z}_q$. Use $p_i H_i^{(1)}$ and $p_i H_i^{(2)}$ to compute

$$p_i H_i^{(1)} \cdot [\![\omega]\!]_1 + p_i H_i^{(2)} \cdot [\![\omega']\!]_1 = p_i \cdot \left(\omega H_i^{(1)} \cdot g_1 + \omega' H_i^{(2)} \cdot g_1\right) = p_i \cdot (\omega \mathbf{h}_{i,1} + \omega' \mathbf{h}_{i,2}) \ .$$

For each $j \in \mathsf{S}$, compute

$$\mathbf{c}_{i,j} = \sigma_{i,j} \cdot \mathbf{f}_1 - j \cdot \sigma_{i,j} \cdot \mathbf{f}_2 + \psi_i \cdot \mathbf{f}_3 = (\sigma_{i,j} \cdot (1, -j),\ \psi_i,\ 0,\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}}$$

where $\sigma_{i,j} \xleftarrow{\$} \mathbb{Z}_q$. Finally, compute

$$\mathbf{t}_i = s_i \cdot [\![\omega]\!]_1 + u_i \cdot [\![\omega']\!]_1 + [\![x_i]\!]_1 = [\![\omega \cdot s_i + \omega' \cdot u_i + x_i]\!]_1$$
$$\mathbf{c}_{i,\mathsf{ipfe}} = p_i \cdot (\omega \cdot \mathbf{h}_{i,1} + \omega' \cdot \mathbf{h}_{i,2}) + \psi_i \cdot \mathbf{h}_{i,3} = (\omega p_i,\ \omega' p_i,\ \psi_i,\ 0)_{\mathbf{H}_i}$$

and output $\mathsf{ct}_{\mathsf{tag},i} \coloneqq \left((\mathbf{c}_{i,j})_j, \mathbf{t}_i, \mathbf{c}_{i,\mathsf{ipfe}}\right)$.

$\underline{\mathsf{Dec}(\mathsf{dk}_{\mathbb{A},\mathbf{y}}, \mathbf{c} \coloneqq (\mathsf{ct}_{\mathsf{tag},i}))}$**:** Parse $\mathsf{ct}_{\mathsf{tag},i} = ((\mathbf{c}_{i,j})_{j\in\mathsf{S}}, \mathbf{t}_i, \mathbf{c}_{i,\mathsf{ipfe}})$ and $\mathsf{dk}_{\mathbb{A},\mathbf{y}} \coloneqq ((\mathbf{k}_{i,j})_{i\in[n], j\in\mathsf{List\text{-}Att}(\mathbb{A})}, (\mathbf{m}_i, \mathbf{k}_{i,\mathsf{ipfe}})_{i\in[n]})$.
If there exists $A \subseteq \mathsf{S}$ and $A \in \mathbb{A}$, then compute the reconstruction vector $\mathbf{c} = (c_j)_j$ of the LSSS for A and

$$[\![\mathsf{out}]\!]_\mathsf{t} = \sum_{i=1}^{n} \left(\left(\sum_{j\in A} \mathbf{c}_{i,j} \times (c_j \cdot \mathbf{k}_{i,j})\right) - (\mathbf{c}_{i,\mathsf{ipfe}} \times \mathbf{k}_{i,\mathsf{ipfe}}) + \mathbf{e}(\mathbf{t}_i, \mathbf{m}_i)\right)$$

Finally, compute the discrete logarithm and output the small value out.

Fig. 3: The construction for multi-client IPFE with fine-grained access control via LSSS from Section 5.2.

Theorem 14. *Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ be a multi-client IPFE scheme with fine-grained access control via LSSS for the functionality class $\mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$, constructed in Section 5.2 in a bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, \mathbf{e}, q)$. Then, $\mathcal{E}$ is one-time IND-secure if the SXDH assumption holds for $\mathbb{G}_1$ and $\mathbb{G}_2$. More specifically, for $\lambda \in \mathbb{Z}$ and for any adversary $\mathcal{A}$, let K denote the total number of functional key queries, P denote the total number of attributes used by $\mathcal{A}$, and Q denote the maximum number of random oracle (RO) queries. We have the following bound:*

$$\mathsf{Adv}^{\mathsf{mc\text{-}ind\text{-}cpa\text{-}1\text{-}chal}}_{\mathcal{E},\mathcal{F}^{\mathsf{IP}},\mathsf{LSSS},\mathcal{A}}(1^\lambda) \leq (2KP \cdot (6P+3) + 2K + 2Q + 5) \cdot \mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda)$$

where $\mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda)$ denotes the maximum advantage over ppt adversaries against the SXDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$ set up with parameter λ.

By combining with Lemma 13, we have the following Corollary:

Corollary 15. *Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ be a multi-client IPFE scheme with fine-grained access control via LSSS, for the functionality class $\mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$, constructed in Section 5.2 in a bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, \mathbf{e}, q)$. Then, $\mathcal{E}$ is IND-secure if the SXDH assumption holds for $\mathbb{G}_1$ and $\mathbb{G}_2$.*

Proof (of Theorem 14 - Main ideas). Recall that in the security proof for single-client adaptive security (Theorem 7) we switch the ℓ-th functional key to semi-functional by augmenting the dimension of the dual bases so that the challenge ciphertext is masked by $\tau \Delta\mathbf{x}[i]$, facing the mask $r_0^{(\ell)}\mathbf{y}^{(\ell)}[i]$ in the corresponding coordinate of the ℓ-th key and $\tau, r_0^{(\ell)} \xleftarrow{\$} \mathbb{Z}_q$ where $\Delta\mathbf{x} := \mathbf{x}_1^* - \mathbf{x}_0^*$. Afterwards, when doing the product of vectors in the dual bases, there will exist the quantity $\sum_{i=1}^n \tau r_0^{(\ell)} \Delta\mathbf{x}[i]\mathbf{y}^{(\ell)}[i] = \tau r_0^{(\ell)} \langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle$, which is non-zero when $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle \neq 0$. The dual bases now must have dimension at least n in order to accommodate all the n terms $\Delta\mathbf{x}[i]\mathbf{y}[i]$. However, in the multi-client setting, we are already using n different dual basis pairs $(\mathbf{H}_i, \mathbf{H}_i^*)$ for n clients and the correctness of the construction in Section 5.2 makes sure that only when gathering all n ciphertext parts can we decrypt to obtain the inner product. Therefore, it suffices to introduce only $\tau_i \Delta\mathbf{x}[i]$ in the component $\mathbf{c}_{i,\mathsf{ipfe}}$ returned from $\mathbf{LoR}$ of client i and only $r_{i,0}^{(\ell)}\mathbf{y}^{(\ell)}[i]$ in the corresponding key component $\mathbf{k}^*_{i,\mathsf{ipfe}}$, while duplicating the pair of bases $(\mathbf{H}_i, \mathbf{H}_i^*)$ for each $i \in [n]$. Indeed, this is also the best we can do because a client i is not supposed to know other inputs $\mathbf{x}_b^*[j]$ of other clients j, where $b \xleftarrow{\$} \{0,1\}$ is the challenge bit. At the same time, we compress the components of the access control part $(\mathbf{c}_{i,j})_j, (\mathbf{k}^*_{i,j})_j$ into the same pair of bases $(\mathbf{F}, \mathbf{F}^*)$ for all clients i. We refer to the introduction for more intuition on this duplicate-and-compress process.

There are some further technical tweaks to be done when applying Lemma 4. First of all, we need the factors $\tau_i, r_{i,0}^{(\ell)}$ to be the same, for the grouping later when doing products of vectors in DPVS. This can be done by using the same $\tau_i = \tau$ for all i and during the basis change to mask the ciphertext component there will be a factor $\Delta\mathbf{x}[i]$. Our argument to introduce $r_{i,0}^{(\ell)}$ in fact does not depend on i and therefore we can use the same $r_{i,0}^{(\ell)} = r_0^{(\ell)}$ for all i as well. One might wonder

if the dependence of the masks still relies on $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle$ because the adversary is not supposed to query **LoR** for corrupted clients and we can only introduce the masks in the vector components of honest i. As a result, the product of vectors in the dual bases in the end will have $\sum_{i \in \mathcal{H}} \tau r_0^{(\ell)} \Delta\mathbf{x}[i]\mathbf{y}^{(\ell)}[i]$. However, the security model imposes that for all corrupted i, the challenge message satisfies $\mathbf{x}_1^*[i] = \mathbf{x}_0^*[i]$ and consequently, $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle = 0$ if and only if $\sum_{i \in \mathcal{H}} \Delta\mathbf{x}[i]\mathbf{y}^{(\ell)}[i] = 0$. This implies that the mask $\tau r_0^{(\ell)} \sum_{i \in \mathcal{H}} \Delta\mathbf{x}[i]\mathbf{y}^{(\ell)}[i]$ persists only when $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle \neq 0$, which is our goal. The masking of ciphertext and key components results from the application of Lemma 4 as we are in the adaptive setting and not knowing what policy the ciphertext's attributes will satisfy. The lemma will mask all vectors $\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$ with $a_0'^{(\ell)} \xleftarrow{\$} \mathbb{Z}_q$, using which we perform a random labeling, and under the constraint that all clients i use the same S, the mask $a_0'^{(\ell)}$ will either appear for all i or neither. This enables us to replace it with $r_0^{(\ell)}$, similarly to the *all-at-once-changing* step in the adaptive single-client proof. We recall that currently the constraint on using the same S for all i is guaranteed by hashing $(\mathsf{tag}, \mathsf{S})$ together. The more complicated and flexible case with possibly different S_i for each i is discussed in Section 5.4. The application of Lemma 4 needs some auxiliary vectors in the dual bases $(\mathbf{F}, \mathbf{F}^*)$, which are not needed in the real usage of the scheme. Following the terminology of Okamoto-Takashima [37], those auxiliary vectors form a *hidden* part of the bases.

The final steps are to change (s_i, u_i) in the challenge ciphertext to (s_i', u_i') so that the ciphertext from **LoR** is encrypting $\mathbf{x}_0^*$ instead of $\mathbf{x}_b^*$ by solving a linear system for $(\Delta S, \Delta U)$ depending on $\mathbf{x}_b^* - \mathbf{x}_0^*$. We stress that the simulation of corrupted keys can still be done using (s_i, u_i) regardless of the order of **LoR** query, under the admissibility from condition 3 in Definition 9 that requires $\Delta\mathbf{x}[i] = \mathbf{x}_1^*[i] - \mathbf{x}_0^*[i] = 0$ if i is corrupted.

In the case of $\langle \Delta\mathbf{x}, \mathbf{y} \rangle \neq 0$, which then implies $\mathbb{A}(\mathsf{S}) \neq 0$, the functional key queries that are simulated using $(\langle S, \mathbf{y} \rangle, \langle U, \mathbf{y} \rangle)$ are computaionally indistinguishable from the ones in correct forms using $(\langle S', \mathbf{y} \rangle, \langle U', \mathbf{y} \rangle)$, under the SXDH assumption. However, the situation is more complicated than the single-client construction because the oracle **Enc** is using (s_i, u_i) as well. In order to be able to perform the correction step on the functional key, we have to program the full-domain hash function, which is modeled as an RO, such that for all queries $(\mathsf{tag}', \mathsf{S}')$ different from the challenge $(\mathsf{tag}, \mathsf{S})$, the value $\mathsf{H}(\mathsf{tag}', \mathsf{S}')$ belongs to $\mathrm{span}([\![(1,\ \mu)]\!]_1) \subseteq \mathbb{G}_1^2$, for $\mu \xleftarrow{\$} \mathbb{Z}_q$. For the challenge $(\mathsf{tag}, \mathsf{S})$, the value $\mathsf{H}(\mathsf{tag}, \mathsf{S})$ remains a pair of random group elements. The main reason behind this is that our correction step requires $\mathsf{H}(\mathsf{tag}', \mathsf{S}')$ belongs to $\mathrm{span}([\![(1,\ \mu)]\!]_1)$ so that it will not affect the normal ciphertext returned from **Enc**. This implies a linear relation between $\Delta S := S' - S$ and $\Delta U := U' - U$. However, if we put $\mathsf{H}(\mathsf{tag}, \mathsf{S})$ on the line $\mathrm{span}([\![(1,\ \mu)]\!]_1)$ as well, then the intention to switch from $\mathbf{x}_0^*$ to $\mathbf{x}_b^*$ in the ciphertext from **LoR** will create another linear relation, which reduces significantly the degree of freedom to choose $(\Delta S, \Delta U)$ in order to make the simulation successful. In the end, the challenge ciphertext no longer depends on b and the advantage becomes 0, concluding the proof. □

5.4 Revisiting MIFE in the Standard Model

We recall that currently our MCFE scheme from Section 5.2 enforces the same $(\mathsf{tag}, \mathsf{S})$ when encrypting for all client $i \in [n]$, by hashing them using the full-domain hash function that is modeled as an RO in the security proof. In practice, this could render a significant cost for synchronisation among clients so as to agree on tag *and* the attributes S at the time of encryption. In addition, by fixing one public tag, one can only obtain an MIFE scheme whose security can be proven in the ROM because we still need the random oracle to process S.

If we allow different $(\mathsf{tag}, \mathsf{S}_i)$ for each client i and during encryption the input for hashing depends only on tag, i.e. $[\![(\omega_{\mathsf{tag}}, \omega'_{\mathsf{tag}})]\!]_1 \leftarrow \mathsf{H}(\mathsf{tag})$, there is a mix-and-match attack among functional keys that has to be considered. More precisely, suppose for two clients $\{1, 2\}$ encrypting $\mathbf{x} = (x_1, x_2)$ under different sets $(\mathsf{S}_1, \mathsf{S}_2)$ of attributes, the ℓ-th and ℓ'-th key queries have access structures $\mathbb{A}$ and $\mathbb{A}'$ where $\mathbb{A}(\mathsf{S}_1) = \mathbb{A}'(\mathsf{S}_2) = 1$ and $\mathbb{A}'(\mathsf{S}_1) = \mathbb{A}(\mathsf{S}_2) = 0$, for the same inner-product with $\mathbf{y} = \mathbf{y}' = (y_1, y_2)$. Neither of these keys should decrypt $x_1y_1 + x_2y_2$ for the sake of security. However, the construction from Figure 3 permits an adversary to use the vectors $\{(\mathbf{c}_{1,j})_j, (\mathbf{k}_{1,j})_j, \mathbf{c}_{1,\mathsf{ipfe}}, \mathbf{k}_{1,\mathsf{ipfe}}\}$ to recover $p_1\omega_{\mathsf{tag}}\langle S, \mathbf{y}\rangle + p_1\omega'_{\mathsf{tag}}\langle U, \mathbf{y}\rangle$. Similar computation allows the same adversary to obtain $p_2\omega_{\mathsf{tag}}\langle S, \mathbf{y}\rangle + p_2\omega'_{\mathsf{tag}}\langle U, \mathbf{y}\rangle$ using $\{(\mathbf{c}_{2,j})_j, (\mathbf{k}_{2,j})_j, \mathbf{c}_{2,\mathsf{ipfe}}, \mathbf{k}_{2,\mathsf{ipfe}}\}$. Finally, observing that $p_1 + p_2 = 1$, exploiting the linear combination $y_1 \cdot [\![\omega_{\mathsf{tag}}s_1 + \omega'_{\mathsf{tag}}u_1 + x_1]\!]_1 + y_2 \cdot [\![\omega_{\mathsf{tag}}s_2 + \omega'_{\mathsf{tag}}u_2 + x_2]\!]_1$ permits finding $\langle \mathbf{x}, \mathbf{y}\rangle$. This demonstrates the main reason why we put S as part of the input to the hash function H in our current scheme. The core of the above problem is the fact that the construction from Section 5.2 does not prohibit combining different "root" vectors $\mathbf{k}_{1,\mathsf{ipfe}}$ and $\mathbf{k}_{2,\mathsf{ipfe}}$ w.r.t different access structure $\mathbb{A}$ and $\mathbb{A}'$.

In this section we present a solution, with minimal modifications to the scheme, to overcome the need for hashing S. Suppose now we are in the more flexible setting where $[\![(\omega_{\mathsf{tag}}, \omega'_{\mathsf{tag}})]\!]_1 \leftarrow \mathsf{H}(\mathsf{tag})$ during encryption. During setup phase, the pair $(\mathbf{H}_i, \mathbf{H}_i^*)$ is a pair of dual bases for $(\mathbb{G}_1^5, \mathbb{G}_2^5)$, with one more dimension compared to our less flexible construction. The master secret key msk stays the same, while the encryption key ek_i now contains furthermore $\theta_i\mathbf{h}_{i,5}$ for some $\theta_i \xleftarrow{\$} \mathbb{Z}_q$. Given an LSSS-realizable monotone access structure $\mathbb{A}$, the key extraction $\mathsf{Extract}(\mathsf{msk}, \mathbb{A}, \mathbf{y} \in \mathbb{Z}_q^n)$ returns $\mathsf{dk}_{\mathbb{A},\mathbf{y}} := ((\mathbf{k}_{i,j})_{i,j}, (\mathbf{m}_i, \mathbf{k}_{i,\mathsf{ipfe}})_{i\in[n]})$. The encryption $\mathsf{Enc}(\mathsf{ek}_i, x_i, \mathsf{tag}, \mathsf{S}_i)$ returns $\mathsf{ct}_{\mathsf{tag},i} := ((\mathbf{c}_{i,j})_j, \mathbf{t}_i, \mathbf{c}_{i,\mathsf{ipfe}})$ for each $i \in [n]$. There is a new element $d_{\mathbb{A},i}$ appearing in the extra coordinate in $\mathbf{k}_{i,\mathsf{ipfe}}$ for every $i \in [n]$, where $(d_{\mathbb{A},i})_i$ satisfies $\sum_{i=1}^n \theta_i d_{\mathbb{A},i} = 0$, independently chosen for each functional keys. The vectors are essentially the same as in Figure 3, except $(\mathbf{c}_{i,\mathsf{ipfe}}, \mathbf{k}_{i,\mathsf{ipfe}})$ for each i as follows:

$$
\begin{aligned}
\mathsf{ek}_i &:= (s_i,\ u_i,\ p_i \cdot H_i^{(1)},\ p_i \cdot H_i^{(2)},\ \mathbf{h}_{i,3},\ \theta_i\mathbf{h}_{i,5},\ \mathbf{f}_1,\ \mathbf{f}_2,\ \mathbf{f}_3)\\
\mathsf{msk} &:= (S,\ U,\ (\theta_i)_i,\ \mathbf{f}_1^*,\ \mathbf{f}_2^*,\ \mathbf{f}_3^*,\ (\mathbf{h}_{i,1}^*, \mathbf{h}_{i,2}^*, \mathbf{h}_{i,3}^*)_{i\in[n]})\\
\mathbf{c}_{i,\mathsf{ipfe}} &:= (\omega_{\mathsf{tag}}p_i,\ \omega'_{\mathsf{tag}}p_i,\ \psi_i,\ 0,\ \theta_i)_{\mathbf{H}_i}\\
\mathbf{k}_{i,\mathsf{ipfe}} &:= (\langle S, \mathbf{y}\rangle, \langle U, \mathbf{y}\rangle,\ a_{i,0} \cdot z,\ 0,\ d_{\mathbb{A},i})_{\mathbf{H}_i^*}
\end{aligned}
$$

The decryption calculation stays invariant because $\sum_{i=1}^{n} \theta_i d_{\mathbb{A},i} = 0$. In retrospection, the mix-and-match attack we gave at the beginning of this section no longer works, because $\mathbb{A} \neq \mathbb{A}'$ and $\theta_1 d_{\mathbb{A},1} + \theta_2 d_{\mathbb{A}',2} = 0$ only with negligible probability over the choices of $\theta_1, \theta_2, d_{\mathbb{A},1}, d_{\mathbb{A}',2} \xleftarrow{\$} \mathbb{Z}_q$, for two independent random families $(d_{\mathbb{A},i})_{i\in[2]}$ and $(d_{\mathbb{A}',i})_{i\in[2]}$. More formally, the security proof for this modified scheme, where we exploit the one extra 5-th coordinate in $(\mathbf{H}_i, \mathbf{H}_i^*)$, can be obtained with recourse to the proof of theorem 14 in section 5.2 under few changes. We sketch the proof and highlight the main differences compared to the less flexible scheme in the full version [34].

Remark 16. Adding this new layer of masking increases the ciphertext's size by only a factor linear in n. Moreover, given this new construction where the set of attributes does not involve in the computation of the full-domain hashing anymore, we can obtain an MIFE in the standard model by fixing one tag for every ciphertext. The random oracle can be removed by publishing a random fixed value corresponding to $\mathsf{H}(\mathsf{tag})$ for encryption. In the end, we obtain an attribute-based MIFE for inner-products with adaptive security in the standard model, where the adversary can make the challenge query to **LoR** at most once for each slot $i \in [n]$. To achieve security w.r.t multiple queries for same slot, we can apply the technique in [21] to enhance our construction with repetitions. Finally, we can apply a layer of All-or-Nothing Encapsulation to the ciphertext components of construction in Section 5.4, so as to remove the tradeoff with respect to partial challenge ciphertexts in case of $(\mathsf{tag}, \mathsf{S}_i)$ for different S_i.

Acknowledgments

We thank Romain Gay for insightful discussions regarding their constructions in [4]. This work was supported in part by the European Union Horizon 2020 ERC Programme (Grant Agreement no. 966570 – CryptAnalytics) and the French ANR Project ANR-19-CE39-0011 PRESTO.

References

1. Abdalla, M., Bellare, M., Neven, G.: Robust encryption. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 480–497. Springer, Heidelberg (Feb 2010). https://doi.org/10.1007/978-3-642-11799-2_28
2. Abdalla, M., Bourse, F., De Caro, A., Pointcheval, D.: Simple functional encryption schemes for inner products. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 733–751. Springer, Heidelberg (Mar / Apr 2015). https://doi.org/10.1007/978-3-662-46447-2_33
3. Abdalla, M., Bourse, F., De Caro, A., Pointcheval, D.: Better security for functional encryption for inner product evaluations. Cryptology ePrint Archive, Report 2016/011 (2016), https://eprint.iacr.org/2016/011
4. Abdalla, M., Catalano, D., Gay, R., Ursu, B.: Inner-product functional encryption with fine-grained access control. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part III. LNCS, vol. 12493, pp. 467–497. Springer, Heidelberg (Dec 2020). https://doi.org/10.1007/978-3-030-64840-4_16

5. Agrawal, S., Kitagawa, F., Modi, A., Nishimaki, R., Yamada, S., Yamakawa, T.: Bounded functional encryption for turing machines: Adaptive security from general assumptions. Cryptology ePrint Archive, Report 2022/316 (2022), https://ia.cr/2022/316
6. Agrawal, S., Libert, B., Stehlé, D.: Fully secure functional encryption for inner products, from standard assumptions. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part III. LNCS, vol. 9816, pp. 333–362. Springer, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53015-3_12
7. Agrawal, S., Maitra, M., Vempati, N.S., Yamada, S.: Functional encryption for turing machines with dynamic bounded collusion from LWE. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part IV. LNCS, vol. 12828, pp. 239–269. Springer, Heidelberg, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84259-8_9
8. Ananth, P., Sahai, A.: Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In: Coron, J.S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part I. LNCS, vol. 10210, pp. 152–181. Springer, Heidelberg (Apr / May 2017). https://doi.org/10.1007/978-3-319-56620-7_6
9. Attrapadung, N., Libert, B., de Panafieu, E.: Expressive key-policy attribute-based encryption with constant-size ciphertexts. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 90–108. Springer, Heidelberg (Mar 2011). https://doi.org/10.1007/978-3-642-19379-8_6
10. Baltico, C.E.Z., Catalano, D., Fiore, D., Gay, R.: Practical functional encryption for quadratic functions with applications to predicate encryption. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 67–98. Springer, Heidelberg (Aug 2017). https://doi.org/10.1007/978-3-319-63688-7_3
11. Beimel, A.: Secure Schemes for Secret Sharing and Key Distribution. Ph.D. thesis, Technion - Israel Institute of Technology, Haifa, Israel (June 1996), https://www.cs.bgu.ac.il/~beimel/Papers/thesis.pdf
12. Bellare, M., O'Neill, A.: Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general definition. In: Abdalla, M., Nita-Rotaru, C., Dahab, R. (eds.) CANS 13. LNCS, vol. 8257, pp. 218–234. Springer, Heidelberg (Nov 2013). https://doi.org/10.1007/978-3-319-02937-5_12
13. Benaloh, J.C., Leichter, J.: Generalized secret sharing and monotone functions. In: Goldwasser, S. (ed.) CRYPTO'88. LNCS, vol. 403, pp. 27–35. Springer, Heidelberg (Aug 1990). https://doi.org/10.1007/0-387-34799-2_3
14. Benhamouda, F., Bourse, F., Lipmaa, H.: CCA-secure inner-product functional encryption from projective hash functions. In: Fehr, S. (ed.) PKC 2017, Part II. LNCS, vol. 10175, pp. 36–66. Springer, Heidelberg (Mar 2017). https://doi.org/10.1007/978-3-662-54388-7_2
15. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (Aug 2001). https://doi.org/10.1007/3-540-44647-8_13
16. Boneh, D., Gentry, C., Hamburg, M.: Space-efficient identity based encryption without pairings. In: 48th FOCS. pp. 647–657. IEEE Computer Society Press (Oct 2007). https://doi.org/10.1109/FOCS.2007.64
17. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (Mar 2011). https://doi.org/10.1007/978-3-642-19571-6_16
18. Castagnos, G., Laguillaumie, F., Tucker, I.: Practical fully secure unrestricted inner product functional encryption modulo p. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018, Part II. LNCS, vol. 11273, pp. 733–764. Springer, Heidelberg (Dec 2018). https://doi.org/10.1007/978-3-030-03329-3_25

19. Chen, J., Lim, H.W., Ling, S., Wang, H., Wee, H.: Shorter IBE and signatures via asymmetric pairings. In: Abdalla, M., Lange, T. (eds.) PAIRING 2012. LNCS, vol. 7708, pp. 122–140. Springer, Heidelberg (May 2013). https://doi.org/10.1007/978-3-642-36334-4_8
20. Chotard, J., Dufour Sans, E., Gay, R., Phan, D.H., Pointcheval, D.: Decentralized multi-client functional encryption for inner product. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018, Part II. LNCS, vol. 11273, pp. 703–732. Springer, Heidelberg (Dec 2018). https://doi.org/10.1007/978-3-030-03329-3_24
21. Chotard, J., Dufour Sans, E., Gay, R., Phan, D.H., Pointcheval, D.: Multi-client functional encryption with repetition for inner product. Cryptology ePrint Archive, Report 2018/1021 (2018), https://eprint.iacr.org/2018/1021
22. Chotard, J., Dufour-Sans, E., Gay, R., Phan, D.H., Pointcheval, D.: Dynamic decentralized functional encryption. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part I. LNCS, vol. 12170, pp. 747–775. Springer, Heidelberg (Aug 2020). https://doi.org/10.1007/978-3-030-56784-2_25
23. Cocks, C.: An identity based encryption scheme based on quadratic residues. In: Honary, B. (ed.) 8th IMA International Conference on Cryptography and Coding. LNCS, vol. 2260, pp. 360–363. Springer, Heidelberg (Dec 2001)
24. Delerablée, C., Gouriou, L., Pointcheval, D.: Key-policy abe with delegation of rights. Cryptology ePrint Archive, Report 2021/867 (2021), https://ia.cr/2021/867
25. Escala, A., Herold, G., Kiltz, E., Ràfols, C., Villar, J.: An algebraic framework for Diffie-Hellman assumptions. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 129–147. Springer, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40084-1_8
26. Gay, R.: A new paradigm for public-key functional encryption for degree-2 polynomials. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020, Part I. LNCS, vol. 12110, pp. 95–120. Springer, Heidelberg (May 2020). https://doi.org/10.1007/978-3-030-45374-9_4
27. Goldwasser, S., Gordon, S.D., Goyal, V., Jain, A., Katz, J., Liu, F.H., Sahai, A., Shi, E., Zhou, H.S.: Multi-input functional encryption. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 578–602. Springer, Heidelberg (May 2014). https://doi.org/10.1007/978-3-642-55220-5_32
28. Gordon, S.D., Katz, J., Liu, F.H., Shi, E., Zhou, H.S.: Multi-input functional encryption. Cryptology ePrint Archive, Report 2013/774 (2013), https://eprint.iacr.org/2013/774
29. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM CCS 2006. pp. 89–98. ACM Press (Oct / Nov 2006). https://doi.org/10.1145/1180405.1180418, available as Cryptology ePrint Archive Report 2006/309
30. Lai, Q., Liu, F.H., Wang, Z.: New lattice two-stage sampling technique and its applications to functional encryption - stronger security and smaller ciphertexts. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 498–527. Springer, Heidelberg (Oct 2021). https://doi.org/10.1007/978-3-030-77870-5_18
31. Lewko, A.B., Waters, B.: New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 455–479. Springer, Heidelberg (Feb 2010). https://doi.org/10.1007/978-3-642-11799-2_27

32. Libert, B., Titiu, R.: Multi-client functional encryption for linear functions in the standard model from LWE. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019, Part III. LNCS, vol. 11923, pp. 520–551. Springer, Heidelberg (Dec 2019). https://doi.org/10.1007/978-3-030-34618-8_18
33. Lin, H.: Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 599–629. Springer, Heidelberg (Aug 2017). https://doi.org/10.1007/978-3-319-63688-7_20
34. Nguyen, K., Phan, D.H., Pointcheval, D.: Multi-client functional encryption with fine-grained access control. Cryptology ePrint Archive, Report 2022/215 (2022), https://eprint.iacr.org/2022/215
35. Okamoto, T., Takashima, K.: Fully secure functional encryption with general relations from the decisional linear assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (Aug 2010). https://doi.org/10.1007/978-3-642-14623-7_11
36. Okamoto, T., Takashima, K.: Adaptively attribute-hiding (hierarchical) inner product encryption. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 591–608. Springer, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_35
37. Okamoto, T., Takashima, K.: Fully secure unbounded inner-product and attribute-based encryption. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 349–366. Springer, Heidelberg (Dec 2012). https://doi.org/10.1007/978-3-642-34961-4_22
38. Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: Ning, P., De Capitani di Vimercati, S., Syverson, P.F. (eds.) ACM CCS 2007. pp. 195–203. ACM Press (Oct 2007). https://doi.org/10.1145/1315245.1315270
39. Pal, T., Dutta, R.: Attribute-based access control for inner product functional encryption from LWE. In: LATIN 2021 (2021)
40. Sahai, A., Waters, B.R.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (May 2005). https://doi.org/10.1007/11426639_27
41. Shamir, A.: How to share a secret. Communications of the Association for Computing Machinery **22**(11), 612–613 (Nov 1979)
42. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO'84. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (Aug 1984)
43. Waters, B.: Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (Aug 2009). https://doi.org/10.1007/978-3-642-03356-8_36
44. Wee, H.: Broadcast encryption with size $N^{1/3}$ and more from k-lin. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part IV. LNCS, vol. 12828, pp. 155–178. Springer, Heidelberg, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84259-8_6