Decentralizing Traitor Tracing: A Multi-Authority Approach

Rishab Goyal
UW-Madison*

Alex Snyder
UW-Madison†

Saikumar Yadugiri
UW-Madison‡

Abstract

Traitor tracing [Chor-Fiat-Naor; CRYPTO'94] has historically been formalized through the lens of a *single* trusted authority that samples the master keys and that, therefore, can read every ciphertext on its own. This *key escrow* problem makes traditional traitor tracing fundamentally incompatible with end-to-end encrypted broadcast networks. In this work, we study *multi-authority traitor tracing* (MA-TT) [Goyal-Yadugiri; ePrint], a new decentralized model for traitor tracing in which the setup is split across K asynchronous and non-interacting authorities, and a user can decrypt only by combining partial keys from *every* authority. We require that semantic security holds even when an arbitrary set of authorities is corrupted (as long as some honest partial key remains hidden for every user), and that traceability never accuses a user for whom an honest partial key remains hidden.

We design MA-TT by combining any multi-authority attribute-based encryption (MA-ABE) scheme with a new primitive that we introduce, distributed mixed functional encryption (DMFE), a careful decentralization of the mixed functional encryption notion [Goyal-Koppula-Waters; STOC'18]. We construct DMFE from LWE via single-key key-homomorphic private constrained PRFs. Plugging in known MA-ABE schemes, we obtain MA-TT from LWE plus pairings with ciphertext size $K \cdot \mathsf{poly}(\lambda, \log N)$. As feasibility, we also give MA-TT from any PKE (with ciphertexts of size $K \cdot N \cdot \mathsf{poly}(\lambda)$) and from any multi-authority functional encryption (with fully succinct parameters).

*Email: rishab@cs.wisc.edu.
†Email: ajsnyder7@wisc.edu.
‡Email: saikumar@cs.wisc.edu.

Contents

1 Introduction **1**

2 Technical Overview **4**
- 2.1 Why Standard TT Does Not Imply MA-TT . . . 5
- 2.2 Our Approach: Lifting GKW18 to the Multi-Authority Setting . . . 6
- 2.3 MA-PLBE from DMFE and MA-ABE . . . 8
- 2.4 Construction of DMFE from LWE . . . 10
- 2.5 Instantiating MA-PLBE . . . 11

3 Preliminaries **12**
- 3.1 Multi-Authority Attribute Based Encryption . . . 12

4 Multi-Authority Traitor Tracing **13**

5 Multi-Authority Private Linear Broadcast Encryption **15**
- 5.1 Definition . . . 15
- 5.2 Distributed Mixed Functional Encryption . . . 17
- 5.3 Multi-Authority PLBE from MA-ABE and DMFE . . . 19

6 Distributed Mixed Functional Encryption from LWE **24**
- 6.1 Preliminary: Almost Key-Homomorphic Private-Constrained PRF . . . 24
- 6.2 DMFE from PCPRF . . . 25

7 Alternate Constructions for MA-TT **32**
- 7.1 Preliminaries . . . 32
- 7.2 MA-PLBE with Public Key Index Encryption Definition . . . 34
- 7.3 MA-PLBE with Public Key Index Encryption from MA-FE . . . 35
- 7.4 Multi-Authority Index Encryption from IBE . . . 37

A Multi-Authority Traitor Tracing from Multi-Authority PLBE **43**
- A.1 Multi-Authority Decoder Private Linear Broadcast Encryption . . . 43
- A.2 Multi-Authority Traitor Tracing from MA-D-PLBE . . . 45
- A.3 Multi-Authority Decoder PLBE from Multi-Authority PLBE . . . 50

1 Introduction

How does one keep authorized users in an encrypted broadcast network accountable, and prevent them from secretly handing over decryption access to unauthorized third parties? Traitor tracing [CFN94] was introduced to solve this accountability problem and protect against insider threats in encryption systems. Traitor tracing (TT) accomplishes this by handing a special *tracing* capability to the central authority which given oracle access to any "useful" decryption device, the tracing algorithm efficiently recovers the identity of (at least one of) the corrupt user(s) whose secret keys must have been used to construct the device. Starting with the seminal work of Chor, Fiat, and Naor [CFN94], TT has been very well studied [NNL01, BS95, BSW06, BW06, BZ14, NWZ16, GKRW18, GKW18, CVW+18a, GQWW19, GKW19, Zha20, AKYY23, GLW23, Zha25]. It has been deployed in practical applications such as Blu-ray content distribution, and has also found surprising connections to differential privacy [DNR+09].

The Key Escrow Problem in Traitor Tracing. Despite this rich line of work, every existing traitor tracing system, going all the way back to [CFN94], is formalized through the lens of a *single*, central trusted authority. This authority is tasked with sampling the master public key (used for encryption) along with the master secret key (used to issue per-user decryption keys). The problem is that in this standard formulation, the central authority *inherently* possesses the ability to decrypt every ciphertext. Whether one calls this the key escrow problem [Rog15] or a single point of failure, the consequence is the same: traditional traitor tracing is fundamentally incompatible with end-to-end encrypted networks, and with any setting where a single trusted setup party is too strong an assumption. Given NIST's recent push to decentralize cryptographic primitives [BP26] and make them more agile [BCC+25], it is high time we revisit traitor tracing with a similar goal.

The State of the Art and its Limitations. Much to the community's surprise, despite the rising tide of decentralized cryptography, there is significant scarcity of research in traitor tracing in a decentralized trust model. To the best of our knowledge, there are only three recent works [Luo24, BLM+24, CMMS25] that attempt to remedy this in the so-called "ad-hoc" or registration-based model. In these models, every individual samples their own public-secret key pair and posts the public key on a public bulletin board. The advantage of this model is clear. There is no key escrow, since no party other than the user ever sees their secret key. However, this model imposes several practical limitations:

- Every user has to first *register* their public key on a bulletin board (via a PKI), and a public, untrusted curator must accumulate *all* of these user public keys into a short "master" public key (which is what the encryptor finally uses). Each user must additionally pull a short *hint key* from the curator before decrypting, with hints needing periodic re-issuance as new users register. This setup is undesirable in scenarios where no party is willing to run the accumulator, or where users' decryption devices have keys burnt into read-only memory (and therefore cannot pick up updated hints).
- Senders cannot push to such a network unless they first download the latest aggregated master key. This is incompatible with low-power broadcast-only devices that have upload capability but little to no 'receive' bandwidth (e.g., sensor networks that broadcast observations to a base station but do not frequently receive firmware updates to re-calibrate the master key).
- There is no clean fallback for a user whose decryption device is lost or whose secret key is accidentally erased. In the registration-based model, the user must register a fresh public key and leave their old key behind as a permanent "ghost" key in the system.

While the ad-hoc/registration-based model is a great first step towards decentralizing traitor tracing, these limitations rule out a wide swathe of natural deployment scenarios. The motivating question for our work is whether we can decentralize traitor tracing in a model that retains the no-key-escrow guarantee, but sidesteps the heavyweight aggregation/hint machinery of registration. Concretely:

(Q) *Can we design traitor tracing systems that resist key escrow even when the setup itself is corrupted, without paying the cost of registration?*

This work. We initiate a formal and systematic study of *multi-authority traitor tracing* (henceforth, MA-TT), a new decentralization model for traitor tracing that draws inspiration from the classic multi-authority model (originally introduced in the context of attribute-based encryption) [Cha07, CC09, LW11, BCG+17, AGT21, GY24]. MA-TT was recently proposed by Goyal and Yadugiri [GY25], and they left it as an open problem to construct MA-TT.

In an MA-TT system, the setup is split across K authorities, each of whom acts *asynchronously and independently*. Concretely, each authority $i \in [K]$ runs its own setup algorithm to sample its own master public-secret key pair $(\mathsf{mpk}_i, \mathsf{msk}_i)$. There is no coordination among these authorities, and they need not even be aware of each other.[1] To encrypt a message m, an encryptor jointly encrypts under $\{\mathsf{mpk}_i\}_{i\in[K]}$.[2] The j-th user obtains a *partial secret key* $\mathsf{sk}_{i,j}$ from each authority i, and decrypts a ciphertext by combining the partial keys $\{\mathsf{sk}_{i,j}\}_{i\in[K]}$ that all sit under the same user identifier j. Decryption fails if any single partial key is missing or mismatched.

Defining security. Like in standard traitor tracing, we require two properties from MA-TT: *semantic security* and *traceability*. Both properties demand new care in the multi-authority setting, since an adversary can now adaptively corrupt any subset $U \subsetneq [K]$ of authorities (recovering their master secret keys) in addition to making partial-key queries on users.

Semantic security under malicious authorities. Suppose only $\delta = |[K] \setminus U|$ authorities remain honest. We require that semantic security holds as long as, for every user j, at least one partial key from the δ honest authorities has *not* been queried by the adversary. Crucially, this means an adversary may corrupt arbitrarily many partial keys for some users so long as no single user has all of their honest partial keys exposed. This is far stronger than the trivial guarantee that one gets by independently combining K standard TT schemes (we discuss this in detail in Section 2).

Traceability. Even when an adversary develops a useful decryption device, the tracer must efficiently identify the user (or users) whose partial keys were used in building the device, while never accusing a user for whom *any* partial key from an honest authority is still hidden from the adversary. In other words, only those users whose partial keys from *every* honest authority have been corrupted are eligible for accusation.

Together, these provides a substantially stronger security guarantee than any standard TT scheme. Encryptors retain semantic security *even against a malicious setup*, and broadcasters retain a strong accountability guarantee against users that maliciously sell or leak their keys. We highlight that no

[1] Following standard practice in the multi-authority literature [LW11], the authorities only share a common reference string (and possibly an a-priori bound on the maximum number of users). Aside from this, each authority's master keys and per-user partial keys are sampled *entirely* from its own local randomness.

[2] The encryptor could actually select any subset of these authorities too.

such hybrid guarantee is provided by any standard/centralized TT system, even those built from heavy hammers such as obfuscation.

Our results. Our main technical contribution is a generic compiler that builds MA-TT from two simpler ingredients:

1. any (statically secure) multi-authority attribute-based encryption (MA-ABE) scheme and
2. a new primitive we introduce, called *distributed mixed functional encryption* (DMFE), which is a careful decentralization of the mixed functional encryption notion [GKW18].

At a high level, this compiler can be viewed as a multi-authority generalization of the elegant ABE plus Mixed-FE template of Goyal, Koppula, and Waters [GKW18] (which itself is a refinement of the Boneh-Sahai-Waters private linear broadcast encryption framework [BSW06]). Critically, both pieces of the compiler need to be in their decentralized incarnation. Any attempt to combine a multi-authority ABE with a single-authority mixed FE, or vice versa, does *not* work, since user partial keys need to be tied across authorities under the same user identifier.

We also give a new construction of DMFE under the LWE assumption, leveraging single-key key-homomorphic private constrained PRFs (KH-PCPRF) [CC17, CVW18b, BTVW17, CVW+18a]. Plugging this together with existing MA-ABE schemes, we obtain the following MA-TT systems, with very different efficiency tradeoffs (also summarized in Table 1):

From LWE and pairings. Combining our DMFE with an MA-ABE scheme for NC^1 from pairings [LW11] (after a packing trick from [GQWW19, AKYY23] to bring DMFE decryption into NC^1), we obtain an MA-TT scheme with *nearly optimal* ciphertext size, $|\mathsf{ct}| = K \cdot \mathsf{poly}(\lambda, \log N, \log K)$. The encryption/decryption time grows as $\mathsf{poly}(N, K)$.

From any MAFE. If one is willing to assume any multi-authority functional encryption [BCG+17, GY24, GY25] for general circuits, then plugging it into a slight specialization of our compiler yields an MA-TT scheme where *all* parameters, $|\mathsf{mpk}|, |\mathsf{sk}|, |\mathsf{ct}|$, and encryption/decryption time are fully succinct: $\mathsf{poly}(\lambda, \log K, \log N)$.

From any PKE. As a feasibility result, we show that any standard public-key encryption scheme suffices to build MA-TT, where the ciphertext size scales with $O(K \cdot N)$. This generalizes the original Chor-Fiat-Naor traitor tracing baseline [CFN94] to the multi-authority setting.

Assumption	$\lvert\mathsf{crs}\rvert$	$\lvert\mathsf{mpk}\rvert$	$\lvert\mathsf{sk}\rvert$	$\lvert\mathsf{ct}\rvert$	Enc	**PT**	**No RO**
PKE	–	N	1	$K \cdot N$	$K \cdot N$	✓	✓
MAFE for P/Poly	–	1	1	1	1	✓	✓
LWE + Pairings	1	1	1	K	$\mathsf{poly}(N, K)$	✗	✗

Table 1: We summarize the sizes of parameters in our MA-TT systems constructed from various assumptions. The columns $|\mathsf{crs}|, |\mathsf{mpk}|, |\mathsf{sk}|, |\mathsf{ct}|$ denote the sizes of the common reference string, the master public key of each authority, the secret key for any user from any authority, and the ciphertext, respectively. Enc denotes the running time of the encryption algorithm. Here, K denotes the number of authorities and N denotes the number of users in the system. Every reported value hides a $\mathsf{poly}(\lambda, \log N, \log K)$ factor, i.e., if the value is reported as X, it should be read as $X \cdot \mathsf{poly}(\lambda, \log N, \log K)$. **PT** represents the feature of *public* traceability of the scheme, and **No RO** denotes the absence of random oracles in the system.

Beyond the headline numbers. We highlight a few qualitative features of our results that are worth emphasizing:

- All of our schemes provide the strong semantic security guarantee discussed above i.e., security holds against *any* number of corrupted authorities, as long as for every user at least one partial key from an honest authority remains hidden.
- The PKE-based and MAFE-based constructions are *publicly traceable*. The LWE-based ones are privately traceable. We leave designing publicly traceable MA-TT from LWE as an interesting open problem.
- The PKE-based and the MAFE-based constructions do not rely on random oracles. The LWE-plus-pairings construction uses random oracles, inherited from the underlying MA-ABE for NC^1 from pairings [LW11].

A new building block of independent interest. The main technical novelty in this work, beyond the lifting of traitor tracing to the multi-authority setting, is the new notion of distributed mixed functional encryption (DMFE) and its construction from LWE. Standard mixed FE already found many applications, outside the context of traitor tracing, in areas such as watermarking and adaptive security [GKM+19, WW24]. We believe DMFE may also be of independent interest beyond multi-authority traitor tracing.

Related work. *Traitor tracing.* There is a vast literature on traitor tracing, starting with Chor-Fiat-Naor [CFN94] and including several seminal works of the last two decades [BSW06, BW06, BZ14, GKRW18, GKW18, CVW+18a, Zha20, AKYY23, GLW23, Zha25]. The community has obtained TT schemes with poly-logarithmic ciphertext size under LWE [GKW18], pairings [Zha25], broadcast-and-trace from standard assumptions [GQWW19, AKYY23], embedded identities [NWZ16, GKW19, Zha25], and a variety of other refinements [Zha20, Zha21, GLW23]. All of these are fundamentally single-authority and inherit the key escrow problem.
Ad-hoc/registration-based TT. The only prior works, to the best of our knowledge, that explicitly study decentralized TT are the registration-based/ad-hoc constructions of Branco et al. [BLM+24], Luo [Luo24], and Chakrabarti et al. [CMMS25]. As discussed above, these works adopt a fundamentally different trust model (registration via PKI plus public curation), which trades the key escrow problem for a different set of practical drawbacks. Our model is incomparable, we keep the multi-authority abstraction (familiar from MA-ABE and MA-FE), and obtain qualitatively different efficiency profiles.
Multi-authority attribute-based and functional encryption. The multi-authority encryption model dates back to Chase [Cha07] and Chase-Chow [CC09]. It has since been extensively developed for ABE (cf. non-exhaustive list [LW11, DKW21, GGL24, WWW22]), inner-product FE [AG21, AGT21, DP23], and most recently for general functional encryption [BCG+17, GY24, GY25]. Our work builds on this foundation and shows that the same multi-authority blueprint is fertile ground for designing traitor tracing.

2 Technical Overview

In this section, we walk through the main ideas behind our results at a high level. We start by defining the notion of multi-authority traitor tracing (MA-TT) in pseudo-syntactic form, together with its desired security properties, and then explain in turn our generic compiler, our new distributed mixed functional encryption (DMFE) building block, and our concrete instantiations.

Defining MA-TT. An MA-TT scheme consists of five algorithms ($\mathsf{GSetup}, \mathsf{ASetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace}$). The global setup algorithm GSetup takes the number of authorities 1^K, and an upper bound 1^N on the number of users, and outputs a common reference string crs which all the remaining algorithms use implicitly. Each authority $i \in [K]$ runs the authority setup algorithm ASetup (independently and asynchronously) sample its own master public-secret key pair $(\mathsf{mpk}_i, \mathsf{msk}_i)$. To issue a partial key to the j-th user for $j \in [N]$, authority i runs $\mathsf{KGen}(i, \mathsf{msk}_i, j)$ and outputs $\mathsf{sk}_{i,j}$. Encrypting message m into a ciphertext ct is done by running $\mathsf{ct} \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m)$. Decryption is done by collecting the $\{\mathsf{sk}_{i,j}\}_{i\in[K]}$ keys (all for the same user $j \in [N]$) and running $\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct})$. Finally, Trace takes oracle access to a useful decryption device D, and a guarded message pair (m_0, m_1), and outputs an accusation set $T \subseteq [N]$. An MA-TT scheme with *private* tracing uses $\{\mathsf{msk}_i\}_{i\in[K]}$ for Trace.

Security. For semantic security, the adversary may (i) adaptively corrupt any set $U \subsetneq [K]$ of authorities and receive their $\{\mathsf{msk}_i\}_{i\in U}$, as well as (ii) query partial keys for arbitrary (i, j) pairs from non-corrupted authorities. The only restriction is that for every user $j \in [N]$, there must remain at least one honest authority $h \in [K] \setminus U$ for which the adversary has *not* queried $\mathsf{sk}_{h,j}$. Under this admissibility, encryptions of m_0 and m_1 must be computationally indistinguishable. For traceability, we require that whenever the adversary outputs a useful decoder D for a subset S, the tracer produces a non-empty accusation set T, all of whose members are users j for which *every* partial key from non-corrupted authorities was queried by the adversary. In particular, no user with at least one hidden partial key may ever be accused.

User Identity	Corrupted Authority 1	Honest Authority 2	Honest Authority 3
User 1	✗	✗	✓
User 2	✗	✗	✗
User 3	✗	✓	✗
User 4	✗	✓	✓

Table 2: Here only user 2 is compromised since all of their partial secret keys either belong to a corrupted authority, or were directly queried by the adversary. User 3 is not compromised because partial secret key for authority 2 and user 3 was not queried by the adversary, and authority 2 was not corrupted.

Desired efficiency. It is highly desirable that the system parameters do *not* grow with N, the number of users, which is consistent with the long line of work on collusion-resistant TT [BSW06, GKW18, CVW+18a, GQWW19, Zha25]. Ideally, one would also like the parameters to be independent of K, the number of authorities. We do not view the second requirement as essential since, in practice, the number of authorities is likely small (think tens or hundreds of state jurisdictions, server admins, officials, etc.), while the number of users could easily be in the tens of thousands even for small-scale broadcast deployments. Consider an average browser which has approximately $K = 200$ tls/ssl certificate authorities that authenticate $N =$ millions of websites.

2.1 Why Standard TT Does Not Imply MA-TT

A reader familiar with traitor tracing might wonder if MA-TT is just a thin generalization of standard TT. You might imagine something like an K-fold parallel composition.

The naïve attempt. Each authority i samples its own traitor tracing system $(\mathsf{mpk}_i, \mathsf{msk}_i)$ independently. To encrypt m, the encryptor K-out-of-K secret shares m into $\{\mathsf{sh}_i\}_{i\in[K]}$, and encrypts

each share sh_i under mpk_i using authority i's TT scheme. A user j's partial key from authority i is simply that authority's standard TT secret key for user j. Decryption is share-by-share recovery and reconstruction.

Mix-and-match attack. Imagine an adversary that obtains the partial key for user 1 and authority 1, the partial key for user 2 and authority 2, ..., and the partial key for user K and authority K. The adversary queries one key per authority, but each under different user identities. The adversary has *not* corrupted any user fully (no single user has been compromised across all authorities), and has *not* corrupted any authority's master secret key. Yet under the naïve scheme, the adversary can clearly decrypt any ciphertext, it has one share's worth of TT key for each of the K shares. Tracing also becomes useless. Each single-authority system is only guaranteed to catch a single corrupted partial key, however, there is no guarantee that these partial keys correspond to the same user j. How do you resolve a conflict where each authority pins the blame on a different user? Are all the blamed users corrupted? Are none of them corrupted? No matter how you resolve these conflicts there will always be attacks that either frame innocent users or allow corrupted users to evade detection.

Takeaway. The whole point of MA-TT is that an adversary that has not fully compromised a user should not be able to frame that user, and an adversary that has not fully corrupted any user should not be able to decrypt a ciphertext. An MA-TT scheme must therefore bind the per-authority partial keys for a user j together in a way that resists any kind of cross-user, cross-authority mix-and-match.

2.2 Our Approach: Lifting GKW18 to the Multi-Authority Setting

At a high level, we adapt the template developed by Goyal, Koppula, and Waters [GKW18] (henceforth referred to as GKW) for constructing standard traitor tracing from lattices, and we lift it to the multi-authority setting. The GKW template is a refinement of the private linear broadcast encryption (PLBE) framework [BSW06], which reduces traitor tracing to a one-message broadcast encryption with N indexed users. In GKW, PLBE is in turn instantiated from two components:

- (ABE) A single-authority attribute-based encryption scheme
- (MFE) A single-authority mixed functional encryption scheme

Roughly, the ABE scheme encrypts each message under a policy that gates decryption on the output of the embedded mixed-FE scheme:

- In the normal (public-key) mode, mixed FE always outputs 1, allowing decryption.
- In the index (secret-key) mode, mixed FE evaluates a comparison predicate $\mathsf{comp}_{\mathsf{index}}(x) = (x \overset{?}{>} \mathsf{index})$, allowing decryption only for users with index above index.

This naturally leads to a tracing strategy via the [BSW06] revoke-and-measure idea. Given a useful decoder box with advantage ϵ in decrypting normal-mode ciphertexts, observe that when we encrypt in index mode with index $\mathsf{index} = N$ no user can decrypt, and so the decoder's advantage drops to 0. Sweeping index from 0 up to N thus produces a sequence of advantages that begins near ϵ and ends near 0, and by the pigeonhole principle there must exist some $j \in [N]$ at which the advantage drops by at least $\epsilon/O(N)$ when we move the index mode ciphertext from j to $j+1$. Such a noticeable drop can only be explained by the decoder possessing a secret key for user i,

who is then accused as a traitor. PLBE executes this idea cleanly by using the above two modes of encryption. The public-key mode that lets all users decrypt, and a secret-key index encryption mode that lets only users with index $j > \mathsf{index}$ decrypt, for some chosen $\mathsf{index} \in [N]$.

Introducing a new tool for realizing MA-TT: MA-PLBE. In this work, we consider the notion of MA-PLBE, the multi-authority analogue of PLBE. An MA-PLBE scheme consists of six algorithms $(\mathsf{GSetup}, \mathsf{ASetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{EncIndex})$, defined analogously to MA-TT, with the $\mathsf{EncIndex}$ algorithm playing the role of index encryption and used during tracing. We require three security properties from MA-PLBE:

Normal hiding: $\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m)$ and $\mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, 0, m)$ are indistinguishable to any attacker, even one that compromises all N users.

Index hiding: for any $i \in [N]$, $\mathsf{EncIndex}(\{\mathsf{msk}_i\}_i, i-1, m)$ and $\mathsf{EncIndex}(\{\mathsf{msk}_i\}_i, i, m)$ are indistinguishable to any attacker, even one that compromises every user other than i.

Message hiding: $\mathsf{EncIndex}(\{\mathsf{msk}_i\}_i, N, m_0)$ and $\mathsf{EncIndex}(\{\mathsf{msk}_i\}_i, N, m_1)$ are indistinguishable to any attacker, even one that compromises all N users.

Crucially, MA-PLBE must satisfy each of these guarantees even when the attacker corrupts arbitrarily many authorities and secret keys, though index hiding is subject to the same compromised-user admissibility condition introduced earlier in the overview.

We show that MA-PLBE suffices for constructing MA-TT, much like in prior works [BSW06, GKW18]. In turn, we construct MA-PLBE from an MA-ABE scheme and a new variant of mixed FE that we call *distributed* mixed FE (DMFE). The distinction between DMFE and *multi-authority* mixed FE, is that in DMFE, the i-th authority issues a partial secret key for the whole input $x = (x_1, \ldots, x_K)$, rather than only its local portion x_i. Decryption then uses a partial key from each authority, all for the same x. This is a strict weakening of the full multi-authority abstraction, but as we will see, it is the right tool for our purposes.

Why not multi-authority mixed FE? A reader familiar with the literature might wonder whether we could instead use multi-authority mixed FE in place of DMFE. In a recent work, [GY25] constructed a private-key MAFE scheme from one-way functions, and it could be possible to combine this with lockable obfuscation [GKW17, WZ17] to yield a multi-authority mixed FE. However, the MAFE scheme of [GY25] only handles a bounded number of corruptions with a large efficiency blow-up. It doesn't appear that any multi-authority FE currently feasible from standard assumptions is good enough for efficient and optimal MA-TT. Moreover, our observation is that full multi-authority mixed FE is overkill for MA-TT. We show that the weaker DMFE notion is sufficient, and we construct it from LWE via the key-homomorphic PCPRF-based mixed FE construction of [CVW+18a], lifted to the distributed setting. Our DMFE construction also achieves optimal parameters, with all sizes $\mathsf{poly}(\lambda, \log N, \log K)$.

Why both halves must be decentralized. One might further ask whether one of the two components could remain single-authority. Neither variant works:

- Pairing parallel single-authority mixed FE schemes with MA-ABE breaks at encryption time. The MA-ABE policy circuit is not hidden from the adversary, so embedding K separate single-authority mixed FE decryption circuits inside it lets the adversary inspect each decryption circuit individually. Using only the mixed FE secret keys from a single authority, the adversary could then learn what user identity the ciphertext is encrypted under.
- Going the other way and placing DMFE inside K parallel secret-shared single-authority ABE schemes brings back the mix-and-match attacks from Section 2.1.

The only setting that works is to have both the ABE part and the mixed FE part live in the *decentralized* world, with the same per-authority $\mathsf{gid} = j$ tying together the partial keys across authorities. This makes DMFE the right tool for MA-TT. It is weak enough to be realizable from standard assumptions such as LWE, yet strong enough to be compiled with MA-ABE into a full MA-PLBE (and hence MA-TT) scheme.

2.3 MA-PLBE from DMFE and MA-ABE

We now show how to construct an MA-PLBE scheme using MA-ABE and DMFE. We begin with recalling standard mixed FE of [GKW18].

Recalling mixed FE. A mixed FE scheme [GKW18] has a central setup that produces $(\mathsf{mpk}, \mathsf{msk})$. Using msk, we can generate secret keys for any input sk_x. There are two encryption modes:

- a public-key mode $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}_i)$. $\mathsf{Dec}(\mathsf{ct}, \mathsf{sk}_x)$ always outputs 1, and
- a secret-key mode $\mathsf{ct} \leftarrow \mathsf{SkEnc}(\mathsf{msk}_i, f)$. $\mathsf{Dec}(\mathsf{ct}, \mathsf{sk}_x)$ outputs $f(x)$.

Function indistinguishability requires that, for any two functions f_0, f_1 that agree on all queried inputs x, $\mathsf{SkEnc}(\mathsf{msk}_i, f_0)$ and $\mathsf{SkEnc}(\mathsf{msk}_i, f_1)$ are computationally indistinguishable. Accept indistinguishability requires that, for any function f that outputs 1 for every queried user key x, $\mathsf{Enc}(\mathsf{mpk}_i)$ is indistinguishable from $\mathsf{SkEnc}(\mathsf{msk}_i, f)$.

Our distributed variant (DMFE). In our distributed mixed FE (DMFE), the setup is decentralized. Each authority i runs its own setup ASetup independently to obtain $(\mathsf{mpk}_i, \mathsf{msk}_i)$. Each authority issues partial secret key for x, $\mathsf{sk}_{i,x}$ using msk_i alone. Note that all partial secret keys are for the same whole input x.

- The public-key encryption mode uses public keys for all authorities, $\mathsf{ct} \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]})$. $\mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct})$ outputs $1, \forall\ x$.
- The secret-key mode encryption $\mathsf{ct} \leftarrow \mathsf{SkEnc}(\{\mathsf{msk}_i\}_{i\in[K]}, f)$ now requires all master secret keys. $\mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct})$ outputs $f(x)$.

We require the same two properties from our DMFE scheme as mixed FE, namely, function and accept indistinguishability. These are defined similarly to mixed FE. The main difference is that these properties should now hold with respect to only those x which are 'compromised'. Only if the attacker possess $\mathsf{sk}_{i,x}$ from every $i \in [K]$, then we consider that query for x to be 'compromised' analogous to a compromised user in MA-TT.

This is a strengthening of function and accept indistinguishability definitions of mixed FE. To see that, consider the function indistinguishability game. In this game, an attacker could still query for partial secret keys for an input x such $f_0(x) \neq f_1(x)$. As long as an attacker does not possess the completed set $\{\mathsf{sk}_{i,x}\}_{i\in[K]}$ (either from corrupted authorities or by partial secret key queries to honest authorities) function indistinguishability should hold. We will now proceed to constructing MA-PLBE via MA-ABE and DMFE.

First Attempt. Once we have DMFE and MA-ABE in hand, it looks like constructing MA-PLBE should follow similarly as GKW. We describe it below.

$\mathsf{ASetup}(i)$**.** Each authority samples an MA-ABE master public-secret key pair $(\mathsf{abe.mpk}, \mathsf{abe.msk})$ and a DMFE master public-secret key pair $(\mathsf{mfe.mpk}, \mathsf{mfe.msk})$. Then $\mathsf{MPK} := (\mathsf{abe.mpk}, \mathsf{mfe.mpk})$ and $\mathsf{MSK} := (\mathsf{abe.msk}, \mathsf{mfe.msk})$

$\mathsf{KGen}(i, \mathsf{MSK}_i, j)$**.** To generate partial secret key for the j-th user, the i-th authority samples $\mathsf{mfe.sk}_{i,j} \leftarrow \mathsf{DMFE.KGen}(\mathsf{mfe.msk}_i, j)$ and then uses MA-ABE to generate $\mathsf{abe.sk}_{i,j} \leftarrow \mathsf{MA\text{-}ABE.KGen}(i, \mathsf{abe.msk}_i, \mathsf{gid} := j, \mathsf{mfe.sk}_{i,j})$. It outputs $\mathsf{SK}_{i,j} := \mathsf{abe.sk}_{i,j}$.

$\mathsf{Enc}(\{\mathsf{MPK}_i\}_{i\in[K]}, m)$ Sample $\mathsf{mfe.ct} \leftarrow \mathsf{DMFE.Enc}(\{\mathsf{mfe.mpk}_i\}_{i\in[K]})$ and then use the DMFE decryption circuit $P := \mathsf{DMFE.Dec}(\cdot, \ldots, \cdot, \mathsf{mfe.ct})$ as policy for MA-ABE and sample $\mathsf{abe.ct} \leftarrow \mathsf{MA\text{-}ABE.Enc}(\{\mathsf{abe.mpk}_i\}_{i\in[K]}, P, m)$. It outputs $\mathsf{CT} := (\mathsf{mfe.ct}, \mathsf{abe.ct})$.

$\mathsf{SkEnc}(\{\mathsf{MSK}_i\}_{i\in[K]}, \mathsf{index}, m)$**.** This is performed similarly to Enc except using the comparison function $\mathsf{comp}_{\mathsf{index}}(x) = \left(x \overset{?}{>} \mathsf{index}\right)$ and $\mathsf{DMFE.SkEnc}$.

Dec**.** This is MA-ABE decryption.

Correctness of the above design from MA-ABE and DMFE correctness. However, there is a major hurdle in proving the security of the above MA-PLBE design.

Message indistinguishability breaks down. Recall that in the message indistinguishability game, we sample $\mathsf{ct} \leftarrow \mathsf{SkEnc}(\{\mathsf{MSK}_i\}_{i\in[K]}, N, m)$ and argue that even if a poly-time attacker compromises all users, the message embedded inside ct remains hidden. In the MA-PLBE scenario, we additionally need to argue even if an attacker corrupts arbitrarily many authorities, this property still holds. This requirement of security against malicious authorities presents a problem in MA-PLBE and this is why above design doesn't work. Let us elaborate.

Consider an attacker in the message indistinguishability game that corrupts partial secret keys $\{\mathsf{SK}_{i,2}\}_{i\in[K]}$ and first authority MSK_1. This is a valid adversary in the MA-PLBE message indistinguishability game as it only corrupted user 2. However, this attacker can still efficiently figure out the message m. This is because the policy $P = \mathsf{DMFE.Dec}(\cdot, \ldots, \cdot, \mathsf{DMFE.ct})$ reveals something non-trivial in presence of authority corruption. Specifically, attacker can find a *bit-string* x'_1 such that $P(x'_1, \mathsf{DMFE.sk}_{2,2}, \ldots, \mathsf{DMFE.sk}_{K,2})$ will be forced to accept. For instance, look at the algebraic structure of our DMFE construction. There is always a way to craft a field element x'_1 which is not a valid DMFE secret key yet it forces decryption to accept. Then attacker can simply generate MA-ABE secret key for authority 1 and user 2 for input x'_1 and learns the encrypted message efficiently. Thus, it looks like we need a stronger notion of DMFE to make the above design of MA-PLBE work.

Getting around MA-ABE authority corruptions. The above issue occurs because we reveal an MA-ABE master secret key for an authority. Our observation is that for tracing in MA-TT, we don't necessarily require the master secret keys of MA-ABE, $\{\mathsf{abe.msk}_i\}_{i\in[K]}$. Indeed as MA-ABE allows one to publicly encrypt any policy and message, we only need $\{\mathsf{DMFE.msk}_i\}_{i\in[K]}$ to craft a secret-key mode ciphertext ct. Keeping this in mind, we change the authority setup algorithm ASetup in the following way: after sampling $\mathsf{DMFE.msk}_i$, sample $\mathsf{DMFE.sk}_{i,j} \leftarrow \mathsf{DMFE.KGen}(\mathsf{DMFE.msk}_i, j)$ and then use $\mathsf{abe.msk}_i$ to sample $\mathsf{SK}_{i,j} \leftarrow \mathsf{MA\text{-}ABE.KGen}(i, \mathsf{abe.msk}_i, \mathsf{gid} := j, \mathsf{DMFE.sk}_{i,j})$ for each $j \in [N]$. This way, we can answer partial secret key generations as well as not leak MA-ABE master secret key.

The normal, index, and message hiding security of our design readily follows from accept indistinguishability of DMFE, function indistinguishability of DMFE, and security of MA-ABE respectively much like prior PLBE designs. We now move to constructing DMFE from LWE.

2.4 Construction of DMFE from LWE

Construction from Key Homomorphic PCPRF. Our DMFE construction follows the [CVW+18a] blueprint for mixed FE, specifically the mixed FE from key-homomorphic private constrained PRFs (PCPRF), defined in Section 6.1 on page 24. At a high level, in their construction, the master secret key is a PCPRF key $\mathsf{prf.msk}$. Secret key for an input x is the PCPRF evaluation at x, $\mathsf{sk}_x = \mathsf{Eval}(\mathsf{prf.msk}, x)$. To encrypt a function f in secret-key mode we constrain $\mathsf{prf.msk}$ for function f to obtain a constrained key $\mathsf{ck} \leftarrow \mathsf{Constrain}(\mathsf{prf.msk}, f)$. By functionality of PCPRFs, we can evaluate the constrained key ck at any x using $\mathsf{CEval}(\mathsf{ck}, x)$. Decryption is simply comparing the constrained evaluation to sk_x. PCPRF's $\mathsf{Constrain}$ method 'hides' the function f in that there is a PPT simulator such that $\mathsf{ck}' \leftarrow \mathsf{SIM}(1^{|f|})$ is indistinguishable from an honestly generated ck. This SIM algorithm doesn't require $\mathsf{prf.msk}$ and this constitutes the public-key mode Enc algorithm. Accept and function indistinguishability properties follow from constrained pseudorandomness property of PCPRF.

In our DMFE scheme, each authority samples its own PCPRF key $\mathsf{prf.msk}_i$ and partial secret key for x: $\mathsf{sk}_{i,x} = \mathsf{Eval}(\mathsf{prf.msk}_i, x)$. In the secret-key mode, we ecrypt a function f under all authority secret keys and combines them via key homomorphism: $\mathsf{prf.msk}^* = \sum_{i\in[K]} \mathsf{prf.msk}_i$. Then we constrain this combined key to compute $\mathsf{ck}^* \leftarrow \mathsf{Constrain}(\mathsf{prf.msk}^*, f)$. Decryption requires $\{\mathsf{sk}_{i,x}\}_{i\in[K]}$, these PCPRF evaluations can then be compared with $\mathsf{CEval}(\mathsf{ck}^*, x)$. If they are close, output is 0 (reject) and if they are far, output is 1 (accept). Once again, public-key mode Enc algorithm simply simulates ck^* using SIM.

Security of DMFE. Function and accept indistinguishability of our DMFE require additional care beyond the single-authority mixed FE setting, since we now need to handle authority corruptions. The natural strategy is for the reduction to guess an honest authority $h \in [K]$, taking a $1/K$ hit in the advantage. Once h is fixed, the reduction samples $\{\mathsf{prf.msk}_i\}_{i\neq h}$ on its own and implicitly sets the PCPRF honest authority's key as $\mathsf{prf.msk}^* - \sum_{i\neq h} \mathsf{prf.msk}_i$, so that the combined key $\mathsf{prf.msk}^* = \sum_{i\in[K]} \mathsf{prf.msk}_i$ used inside SkEnc aligns with the underlying PCPRF challenge. After the adversary declares the challenge function f, the reduction forwards f to its PCPRF challenger and receives a constrained key ck^*, which it uses as the DMFE challenge ciphertext. To answer partial-key queries on inputs x at the honest authority h, the reduction leverages the key homomorphism and functionality preservation of the PCPRF to compute $\mathsf{Eval}(\mathsf{prf.msk}_h, x) \approx \mathsf{CEval}(\mathsf{ck}^*, x) - \mathsf{Eval}\left(\sum_{i\neq h} \mathsf{prf.msk}_i, x\right)$ whenever $f(x) = 0$ (so that CEval roughly matches the real evaluation), and uses a uniformly random response in place of $\mathsf{Eval}(\mathsf{prf.msk}_h, x)$ when $f(x) = 1$, relying on the constraint-hiding property of the PCPRF. The case analysis on $f(x)$ requires some additional care, but using the $1/K$ guessing loss, everything goes through.

Arguing 1-Bounded Security. Following [GKW18], the traceability proof of MA-TT requires a stronger flavor of DMFE security where accept and function indistinguishability continue to hold even when the attacker is given one additional secret-key ciphertext $\mathsf{ct}' \leftarrow \mathsf{SkEnc}(\{\mathsf{prf.msk}_i\}_{i\in[K]}, f')$ beyond the challenge ciphertext. To support this 1-bounded notion, we modify each authority to sample 2λ secret keys $\{\mathsf{prf.msk}_{i,\zeta,b}\}_{\zeta\in[\lambda],b\in\{0,1\}}$ instead of a single $\mathsf{prf.msk}_i$. Each secret-key mode encryption is then associated with a fresh random string $z \in \{0,1\}^\lambda$, and constrains the selected combination $\mathsf{prf.msk}^* = \sum_{i\in[K]} \sum_{\zeta\in[\lambda]} \mathsf{prf.msk}_{i,\zeta,z_\zeta}$. This is the same technique used in [GKW18, CVW+18a, GQWW19] for 1-bounded mixed FE.

Efficiency of DMFE. An interesting feature of this construction is that the parameter sizes do *not* grow with the number of authorities. Each ciphertext is just a single constrained key ck^* whose

size is independent of number of users N and number of authorities K. This yields a fully optimal DMFE scheme from LWE. Looking ahead, the inadequacies in efficiencies of our results occurs due to the MA-ABE schemes from pairings and not because of our compiler. Specifically, if there exists an MA-ABE scheme for TC^1 circuits where ciphertext sizes are succinct (relative to K), we could achieve fully optimal MA-TT. We leave constructing such an MA-ABE scheme as an interesting open problem.

2.5 Instantiating MA-PLBE

We now describe 3 ways to instantiate our MA-PLBE scheme and achieve MA-TT with varying efficiency and functionalities.

(I) LWE + pairings. The most natural starting point is to use the MA-ABE-for-NC^1 scheme of Lewko-Waters [LW11], which use bilinear pairings (and a random oracle). Our challenge here is that our DMFE decryption circuit contains a call to the PCPRF CEval, which is not in NC^1 (it is in TC^1 [GQWW19]). We follow the pre-processing and packing technique of Goyal et al. [GQWW19] (used originally in the context of broadcast-and-trace) to compress the DMFE decryption circuit into NC^1. The packing step, however, blows up the input size from $\mathsf{poly}(\log N, K)$ to $\mathsf{poly}(N^{1/\epsilon}, K)$ for any constant $\epsilon > 0$, which translates into ciphertexts of size $N^{\epsilon} \cdot K \cdot \mathsf{poly}(\lambda)$.

Recovery via the AKYY-style technique. To recover, we employ the technique (first observed in a slightly different context by Agrawal, Kumari, Yadav, and Yamada [AKYY23]) of moving the packing computation to the decryptor's side. As the packing step is deterministic, we keep the DMFE ciphertext un-packed in the MA-TT ciphertext, and let the decryptor itself perform the packing on the fly before invoking MA-ABE decryption. This restores a near-optimal ciphertext size of $K \cdot \mathsf{poly}(\lambda, \log N, \log K)$ while pushing the $\mathsf{poly}(N, K)$ cost into the encryption/decryption running time. Note that we still incur the cost of K in ciphertext size due to existing constructions of MA-ABE for NC^1 circuits having ciphertexts proportional to the number of authorities involved in the encryption. We leave it as an interesting open problem to construct an MA-ABE for NC^1 circuits where ciphertexts are succinct with K as it readily yields an optimal MA-TT scheme albeit with large encryption/decryption running time.

An alternate "faster decryption" route. We can also offer an alternate way of decrypting in the above instantiation where decryption time is independent of N. Here, the encryptor ships the packed DMFE ciphertext in the MA-TT ciphertext. This removes the need for the decryptor to do the packing work, at the cost of a N^{ϵ} blow-up in $|\mathsf{ct}|$.

For completeness, we briefly describe two further constructions that do not go through the DMFE machinery.

(II) From any PKE. As a baseline feasibility result, we show that any standard public-key encryption scheme suffices to build an MA-PLBE scheme with public key index encryption which in turn yields an MA-TT scheme with public traceability. Combining PKE with simple K-out-of-K secret sharing yields an MA-TT scheme[3]. The construction is straightforward, but the ciphertext size scale linearly in $K \cdot N$, mirroring the folklore baseline.

(III) From any MAFE. If we are willing to use a multi-authority functional encryption scheme for general polynomial-size circuits [BCG+17, GY25], then we can directly run the mixed-FE-style

[3]This construction contains large master public keys which grows linearly with number of users N. We can use identity-based encryption (IBE) [Sha85, BF01] to reduce this to a constant size master public key.

index-encryption in the underlying MAFE itself, without any need for DMFE or the ABE-plus-mixed-FE compiler. This gives an MA-TT scheme whose parameters are fully succinct, but at the cost of relying on MAFE for general circuits, which itself a heavy primitive currently only known from sub-exponentially-hard obfuscation [BCG+17] or from LWE / SXDH with the bounded corruptions [GY25]. Notably neither of these schemes are fully succinct relative to K, constructing an MAFE scheme succinct in K (even from iO) is an interesting open problem we leave to the reader.

Open Directions. Our work suggests several natural follow-up questions:

1. Can we obtain fully-succinct ciphertext from standard assumptions (LWE and pairings, without sub-exponential obfuscation)?
2. Can we get an MA-TT from just pairings or just LWE, without relying on both? Can we improve the encryption/decryption time of our LWE+pairings construction below $\mathsf{poly}(N, K)$?
3. We only handle a bounded number of users. Is there an analogue of the embedded-identity TT model [NWZ16, GKW19] in the multi-authority world, where users carry full identities and need not be pre-registered?
4. Can we support richer access structures, such as monotone access policies beyond threshold access structures over an authority subset, so that a user with keys from a quorum of authorities can decrypt?
5. Currently, our traceability is guaranteed only when all authorities behave honestly during Trace. Can we strengthen this to allow corruption of authorities during the traceability game too? Can we trace these corrupted authorities?

These are all interesting questions, and we leave them to future work. Our focus in this paper is on introducing the MA-TT model and showing its feasibility from a wide range of standard assumptions.

3 Preliminaries

Notation. By PPT we denote probabilistic polynomial-time. We denote the security parameter by λ and the set of positive integers by $\mathbb{N}$. For any $a, b \in \{0\} \cup \mathbb{N}$, $a \leq b$, we denote by $[a, b]$, the set of all integers from a to b including a and b. In other words, $[a, b] = \{a, \ldots, b\}$. We denote by $[n] := [1, n]$. We denote by $x \xleftarrow{\$} \mathcal{X}$, the process of sampling an element x from the set $\mathcal{X}$, with uniform probability. Similarly, for any PPT algorithm $\mathcal{A}$, $x \leftarrow \mathcal{A}(y)$ denotes the process of sampling x from the output distribution of $\mathcal{A}$ when run on y. By $\mathsf{negl}(\cdot)$, we denote negligible functions. By poly, we denote positive polynomials. All of our logarithms log are in base 2.

We say that two efficiently samplable probability distributions $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ are computationally indistinguishable if for any non-uniform PPT distinguisher $\mathcal{D} = \{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$, and for large enough $\lambda \in \mathbb{N}$,

$$\left| \Pr_{\alpha\leftarrow\mathcal{X}_\lambda}\left[1 \leftarrow \mathcal{D}(1^\lambda, \alpha)\right] - \Pr_{\alpha\leftarrow\mathcal{Y}_\lambda}\left[1 \leftarrow \mathcal{D}(1^\lambda, \alpha)\right] \right| \leq \mathsf{negl}(\lambda)$$

We denote statistical indistinguishability by $\approx_s$ and identical distributions by $\equiv$.

3.1 Multi-Authority Attribute Based Encryption

Syntax. A Multi-Authority Attribute Based Encryption (MA-ABE) scheme for message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_\lambda$, attribute space $\mathcal{X} = \{\mathcal{X}_\kappa\}_\kappa$ and predicate class $\mathcal{F} = \{\mathcal{F}_{\kappa,K} = \{f : \mathcal{X}_\kappa^K \to \{0, 1\}\}\}_{\kappa,K}$

This scheme consists of the following polynomial time algorithms.

$\mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa) \to \mathsf{crs}$ The randomized global setup algorithm takes as input security parameter λ, number of authorities K, functionality parameter κ, and outputs common reference string crs, which is assumed to be an input of all the following algorithms.

$\mathsf{ASetup}(i) \to (\mathsf{mpk}_i, \mathsf{msk}_i)$ The randomized authority setup algorithm takes as input an authority index $i \in [K]$, and outputs associated master secret key msk_i and master public key mpk_i.

$\mathsf{KGen}(i, \mathsf{msk}_i, x_i, \mathsf{gid}) \to \mathsf{sk}_{i,x_i,\mathsf{gid}}$ The randomized key generation algorithms takes as input an authority index i along with an associated master secret key msk_i, partial attribute x_i, and global identity gid. Then it returns a secret key $\mathsf{sk}_{i,x_i,\mathsf{gid}}$.

$\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, f, m) \to \mathsf{ct}_f$ The randomized encryption algorithm takes as input a master public key mpk_i from each authority index, a policy f, and a message m. It returns a ciphertext ct_f.

$\mathsf{Dec}(\{\mathsf{sk}_{i,x_i,\mathsf{gid}}\}_{i\in[K]}, \mathsf{ct}_f) \to m/\perp$ The deterministic decryption algorithm takes a ciphertext ct_f and a set of secret keys $\mathsf{sk}_{i,x_i,\mathsf{gid}}$ generated by each authority index under the same user identity. Then it either returns the message m or $\perp$.

Definition 3.1. A Multi-Authority Attribute-Based Encryption scheme (GSetup, ASetup, KGen, Enc, Dec) must satisfy the following properties.

Correctness There exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $K = \mathsf{poly}(\lambda)$, $\kappa = \mathsf{poly}(\lambda)$, $m \in \mathcal{M}_\lambda$, $f \in \mathcal{F}_{\kappa,K}$, $\{x_i \in \mathcal{X}_\kappa\}_{i\in[K]}$,
If $f(x_1, ..., x_a) = 1$

$$\Pr\left[\begin{array}{lcl} & & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa) \\ & & \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ m' = m & : & \forall\ i \in [K], \mathsf{sk}_{i,x_i,\mathsf{gid}} \leftarrow \mathsf{KGen}(i, \mathsf{msk}_i, x_i, \mathsf{gid}) \\ & & \mathsf{ct}_f \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, f, m) \\ & & m' \leftarrow \mathsf{Dec}(\{\mathsf{sk}_{i,x_i,\mathsf{gid}}\}_{i\in[K]}, \mathsf{ct}_f) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Static Security For any admissible PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{lcl} & & (1^K, 1^\kappa, f, Q, U) \leftarrow \mathcal{A}, \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa) \\ & & \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ b \leftarrow \mathcal{A}(\mathsf{ct}) & : & \forall\ (i, x_i, \mathsf{gid}) \in Q, \mathsf{sk}_{i,x_i,\mathsf{gid}} \leftarrow \mathsf{KGen}(i, \mathsf{msk}_i, x_i, \mathsf{gid}) \\ & & (m_0, m_1) \leftarrow \mathcal{A}\left(\begin{array}{l} \mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}, \{\mathsf{msk}_i\}_{i\in U}, \\ \{\mathsf{sk}_{i,x_i,\mathsf{gid}}\}_{(i,x_i,\mathsf{gid})\in Q} \end{array}\right) \\ & & b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, f, m_b) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

For $i \notin U$ let $\mathcal{X}_i = \{(x_i, \mathsf{gid}) \mid (i, x_i, \mathsf{gid}) \in Q\}$, for $i \in U$ let $\mathcal{X}_i = \{(x_i, \mathsf{gid}) \mid x_i \in \{0,1\}^*, \mathsf{gid} \in \mathsf{GID}\}$. Let $\mathcal{X}^* = \{(x_1, ..., x_K) \mid (x_i, \mathsf{gid}) \in \mathcal{X}_i \mid \mathsf{gid} \in \mathsf{GID}\}$, then $\mathcal{A}$ cannot query the sets Q, U such that there exists $(x_1, ..., x_K) \in \mathcal{X}^*$ where $f(x_1, ..., x_K) = 1$.

4 Multi-Authority Traitor Tracing

Syntax. A Multi-Authority Traitor Tracing (MA-TT) scheme for message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_\lambda$ consists of the following polynomial time algorithms.

$\mathsf{GSetup}(1^\lambda, 1^K, 1^N) \to \mathsf{crs}$ The possibly randomized global setup algorithm takes as input security parameter λ, number of authorities K, number of users N, and outputs common reference string crs, which is assumed to be an input of all the following functions.

$\mathsf{ASetup}(i) \to (\mathsf{mpk}_i, \mathsf{msk}_i)$ The randomized authority setup algorithm takes as input an authority index $i \in [K]$, and outputs associated master secret key msk_i and master public key mpk_i.

$\mathsf{KGen}(i, \mathsf{msk}_i, j) \to \mathsf{sk}_{i,j}$ The randomized key generation algorithms takes as input an authority index i, master secret key msk_i, and user index $j \in [1, ..., N]$. Then it outputs secret key $\mathsf{sk}_{i,j}$.

$\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m) \to \mathsf{ct}$ The randomized encryption algorithm takes as input a master public key from each authority index, and a message m. It returns a ciphertext ct.

$\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct}) \to m/\perp$ The deterministic decryption algorithm takes a ciphertext ct and a set of secret keys $\mathsf{sk}_{i,j}$, one for each authority index. Then it either returns the message m or $\perp$.

$\mathsf{Trace}^D(\{\mathsf{msk}_i\}_{i\in[K]}, 1^y, m_0, m_1) \to T$ The randomized trace algorithm takes a master secret key msk_i from each authority index. Then it uses oracle access to a decoder box D, and decoder box advantage value $\frac{1}{y}$, along with a set of messages m_0, m_1. It returns a set $T \subset [N]$ of user indexes.

Definition 4.1. A Multi-Authority Traitor Tracing scheme ($\mathsf{GSetup}, \mathsf{ASetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace}$) must satisfy the following properties.

Correctness There exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $K = \mathsf{poly}(\lambda)$, $N = \mathsf{poly}(\lambda)$, $m \in \mathcal{M}_\lambda, j \in [N]$,

$$\Pr\left[\begin{array}{l|l} \mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct}) = m & \begin{array}{l}\mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N) \\ \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ \forall\ i \in [K], \mathsf{sk}_{i,j} \leftarrow \mathsf{KGen}(i, \mathsf{msk}_i, j) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m)\end{array} \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Semantic Security For any admissible PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{l|l} b \leftarrow \mathcal{A}(\mathsf{ct}_b) & \begin{array}{l}(1^K, 1^N) \leftarrow \mathcal{A} \\ \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N) \\ \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ (m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{KGen}(\cdot, \mathsf{msk}_{(\cdot)}, \cdot), \mathsf{Corr}(\cdot)}(\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}) \\ b \leftarrow \{0, 1\} \\ \mathsf{ct}_b \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m_b)\end{array} \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

The corruption oracle Corr takes an input authority index $i \in [K]$ and responds with msk_i. $\mathcal{A}$ cannot query Corr for every authority index.

The KGen oracle takes authority index $i \in [K]$ and user index $j \in [N]$ and responds with a secret key generated under i and j.

Let U be the set of authority indexes queried by the Corr oracle, for every $j \in [N]$ $\mathcal{A}$ cannot query the KGen oracle for all $\{(i, j)\}_{i\in[K]\setminus U}$

IND-Secure Traitor Tracing Consider the following Tracing Experiment on page 15 $\mathsf{Tr}^{\mathcal{T}}_{\mathcal{A},\epsilon}(1^\lambda)$ with Multi-Authority Traitor Tracing scheme $\mathcal{T} = (\mathsf{GSetup}, \mathsf{ASetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$. Parameterized by adversary $\mathcal{A}$, non-negligible function $\epsilon(\cdot)$, and $\lambda \in \mathbb{N}$.

Tracing Experiment

$$(1^K, 1^N) \leftarrow \mathcal{A}$$
$$\mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N)$$
$$\forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i)$$
$$(D, m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{KGen}(\cdot,\mathsf{msk}_{(\cdot)},\cdot),\mathsf{Corr}(\cdot)}(\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]})$$
$$T \leftarrow \mathsf{Trace}^D(\{\mathsf{msk}_i\}_{i\in[K]}, 1^y, m_0, m_1)$$

The corruption oracle Corr takes an input authority index $i \in [K]$ and responds with msk_i. $\mathcal{A}$ cannot query Corr for every authority index.
The KGen oracle takes authority index $i \in [K]$ and user index $j \in [N]$ and responds with a secret key generated under i and j.
Let U be the set of authority indexes queried by the Corr oracle, then let S be the set of all $j \in [N]$ where $\mathcal{A}$ queried the KGen oracle for all $\{(i,j)\}_{i\in[K]\setminus U}$

Based on the Tracing Experiment $\mathsf{Tr}^{\mathcal{T}}_{\mathcal{A},\epsilon}(1^\lambda)$ we are going to define three probabilistic events over the randomness of $\mathcal{T}$.

$$\text{Let } \mathsf{Adv}^D = \Pr\left[\ D(\mathsf{ct}) = b \quad : \quad \begin{array}{l} b \leftarrow \{0,1\} \\ \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m_b) \rightarrow ct \end{array}\ \right] - \frac{1}{2}$$

- $\mathsf{GoodDecoder} : \mathsf{Adv}^D \geq \epsilon(\lambda)$
- $\mathsf{FalseTrace} : T \not\subset S$
- $\mathsf{CorrectTrace} : T \neq \emptyset$

A Multi-Authority Traitor Tracing scheme $\mathcal{T}$ is said to be IND-Secure if for every PPT adversary $\mathcal{A}$ and non-negligible function function $\epsilon(\cdot)$ there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ the following holds.

- $\Pr[\mathsf{CorrectTrace}] \geq \Pr[\mathsf{GoodDecoder}] - \mathsf{negl}(\lambda)$
- $\Pr[\mathsf{FalseTrace}] \leq \mathsf{negl}(\lambda)$

5 Multi-Authority Private Linear Broadcast Encryption

5.1 Definition

Syntax. A Multi-Authority Private Linear Broadcast Encryption (MA-PLBE) scheme for message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_\lambda$ consists of the following polynomial time algorithms.

$\mathsf{GSetup}(1^\lambda, 1^K, 1^N) \to \mathsf{crs}$ The randomized global setup algorithm takes as input security parameter λ, number of authorities K, and outputs common reference string crs, which is assumed to be an input of all the following functions.

$\mathsf{ASetup}(i) \to (\mathsf{mpk}_i, \mathsf{msk}_i)$ The randomized authority setup algorithm takes as input an authority index $i \in [K]$, and outputs associated master secret key msk_i and master public key mpk_i.

$\mathsf{KGen}(i, \mathsf{msk}_i, j) \to \mathsf{sk}_{i,j}$ The randomized key generation algorithms takes as input an authority index i along with an associated master secret key msk_i. Then it uses the inputted user index $j \in [N]$ to generate a user secret key associated with authority i and user j called $\mathsf{sk}_{i,j}$.

$\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m) \to \mathsf{ct}$ The randomized encryption algorithm takes as input a master public key mpk_i from each authority index, and a message m. It returns a ciphertext ct.

$\mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, j, m) \to \mathsf{ct}$ The randomized secret key encryption algorithm takes as input all authority master secret keys msk_i the message m, and a user index $j \in [0, ..., N]$. Returning a ciphertext ct.

$\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct}) \to m/\perp$ The deterministic decryption algorithm takes a ciphertext ct and a set of secret keys $\mathsf{sk}_{i,j}$ generated by each authority index under the same user index j. Then it either returns the message m or $\perp$.

Definition 5.1. A Multi-Authority Private Linear Broadcast Encryption scheme ($\mathsf{GSetup}, \mathsf{ASetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{EncIndex}, \mathsf{Dec}$) must satisfy the following properties.

Correctness There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}, K = \mathsf{poly}(\lambda), N \in \mathsf{poly}(\lambda)$, $m \in \mathcal{M}_\lambda, j \in [N]$, $k \in [0, ..., \lambda]$,

$$\Pr\left[\begin{array}{l|l} \mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct}) = m & \begin{array}{l}\mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N) \\ \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ \forall\ i \in [K], \mathsf{sk}_{i,j} \leftarrow \mathsf{KGen}(i, \mathsf{msk}_i, j) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m)\end{array}\end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

If $j > k$

$$\Pr\left[\begin{array}{l|l} \mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct}) = m & \begin{array}{l}\mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N) \\ \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ \forall\ i \in [K], \mathsf{sk}_{i,j} \leftarrow \mathsf{KGen}(i, \mathsf{msk}_i, j) \\ \mathsf{ct} \leftarrow \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, k, m)\end{array}\end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Normal Hiding Security For any admissible PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{l|l} b \leftarrow \mathcal{A}^{\mathsf{EncIndex}(\{\mathsf{msk}_i\}_i, 0, \cdot)}(\mathsf{ct}_b) & \begin{array}{l}(1^K, 1^N, U) \leftarrow \mathcal{A} \\ \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N) \\ \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ (\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}, \{\mathsf{msk}_i\}_{i\in U}) \to \mathcal{A} \\ m \leftarrow \mathcal{A}^{\mathsf{KGen}(\cdot, \mathsf{msk}_{(\cdot)}, \cdot), \mathsf{EncIndex}(\{\mathsf{msk}_i\}_i, 0, \cdot)} \\ b \leftarrow \{0, 1\} \\ \mathsf{ct}_0 \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m) \\ \mathsf{ct}_1 \leftarrow \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, 0, m)\end{array}\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

The $\mathsf{EncIndex}$ oracle takes a message m and returns a ciphertext encrypting m under user index 0. $\mathcal{A}$ can only make one query to the $\mathsf{EncIndex}$ oracle
There are no restrictions on the use of the KGen oracle.
The KGen oracle responds with a key generated for some authority index $i \in [K]$ and user index $j \in [N]$.

Index Hiding Security For any admissible PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{l} b \leftarrow \mathcal{A}^{\mathsf{EncIndex}(\{\mathsf{msk}_i\}_i,\cdot,\cdot)}(\mathsf{ct}) \quad : \quad \begin{array}{l}(1^K, 1^N, j, U) \leftarrow \mathcal{A} \\ \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N) \\ \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ (\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}, \{\mathsf{msk}_i\}_{i\in U}) \rightarrow \mathcal{A}^{\mathsf{KGen}(\cdot,\mathsf{msk}_{(\cdot)},\cdot),\mathsf{EncIndex}(\{\mathsf{msk}_i\}_i,\cdot,\cdot)} \\ m \leftarrow \mathcal{A} \\ b \leftarrow \{0,1\} \\ \mathsf{ct} \leftarrow \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, j-b, m)\end{array}\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

The $\mathsf{EncIndex}$ oracle takes a user index $j \in [0, ..., N]$ and a message m and returns a ciphertext encrypting m under index j. $\mathcal{A}$ can only make one query to the $\mathsf{EncIndex}$ oracle.
$\mathcal{A}$ cannot query the KGen oracle for every $\{(i,j)\}_{i\in[K]\setminus U}$.

Message Hiding Security For any admissible PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{l} b \leftarrow \mathcal{A}^{\mathsf{EncIndex}(\{\mathsf{msk}_i\}_i,\cdot,\cdot)}(\mathsf{ct}) \quad : \quad \begin{array}{l}(1^K, 1^N, U) \leftarrow \mathcal{A} \\ \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N) \\ \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ (\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}, , \{\mathsf{msk}_i\}_{i\in U}) \rightarrow \mathcal{A}^{\mathsf{KGen}(\cdot,\mathsf{msk}_{(\cdot)},\cdot),\mathsf{EncIndex}(\{\mathsf{msk}_i\}_i,\cdot,\cdot)} \\ (m_0, m_1) \leftarrow \mathcal{A} \\ b \leftarrow \{0,1\} \\ \mathsf{ct} \leftarrow \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, N, m_b)\end{array}\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

The $\mathsf{EncIndex}$ oracle takes a user index $j \in [0, ..., N]$ and a message m and returns a ciphertext encrypting m under index j. $\mathcal{A}$ can only make one query to the $\mathsf{EncIndex}$ oracle.
There are no restrictions on the use of the KGen oracle.

5.2 Distributed Mixed Functional Encryption

Syntax. A Distributed Mixed Functional Encryption (DMFE) scheme for function input space $\mathcal{X} = \{\mathcal{X}_\kappa\}_{\kappa\in\mathbb{N}}$ and function class $F = \{\mathcal{F}_\kappa : \mathcal{X}_\kappa \rightarrow \{0,1\}\}_{\kappa\in\mathbb{N}}$. This scheme consists of the following polynomial time algorithms.

$\mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa) \rightarrow \mathsf{crs}$ The randomized global setup algorithm takes as input security parameter λ, number of authorities K, functionality index κ, and outputs common reference string crs, which is assumed to be an input of all the following functions.

$\mathsf{ASetup}(i) \rightarrow (\mathsf{mpk}_i, \mathsf{msk}_i)$ The randomized authority setup algorithm takes as input an authority index $i \in [K]$, and outputs associated master secret key msk_i and master public key mpk_i.

$\mathsf{KGen}(i, \mathsf{msk}_i, x) \to \mathsf{sk}_{i,x}$ The randomized key generation algorithms takes as input an authority index i along with an associated master secret key msk_i. Then it uses the inputted value x to generate a user secret key associated with authority i and value x called $\mathsf{sk}_{i,x}$.

$\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}) \to \mathsf{ct}$ The randomized public key encryption algorithm takes as input a master public key from each authority index, and returns a ciphertext which encrypts the all-accepting function.

$\mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct}) \to \{0,1\}$ The deterministic decryption algorithm takes a ciphertext ct and a set of secret keys generated by each authority index under the same value. Then it returns a number representing the functional evalution encrypted by the ciphertext on x.

$\mathsf{SkEnc}(\{\mathsf{msk}_i\}_{i\in[K]}, f) \to \mathsf{ct}$ The randomized secret key encryption algorithm takes as input a master secret key from each authority index, and returns a ciphertext which encrypts the function f.

Definition 5.2. A Distributed Mixed-Functional Encryption scheme (GSetup, ASetup, KGen, Enc, Dec, SkEnc) must satisfy the following properties.

Correctness There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$, $K = \mathsf{poly}(\lambda)$, $\kappa = \mathsf{poly}(\lambda)$, $f \in \mathcal{F}_\kappa$, $x \in \mathcal{X}_\kappa$,

$$\Pr\left[\begin{array}{rcl} & & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa) \\ & & \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ b = 1 & : & \forall\ i \in [K], \mathsf{sk}_{i,x} \leftarrow \mathsf{KGen}(i, \mathsf{msk}_i, x) \\ & & \mathsf{ct}_1 \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}) \\ & & b \leftarrow \mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct}_1) \end{array}\right] > 1 - \mathsf{negl}(\lambda)$$

$$\Pr\left[\begin{array}{rcl} & & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa) \\ & & \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ b = f(x) & : & \forall\ i \in [K], \mathsf{sk}_{i,x} \leftarrow \mathsf{KGen}(i, \mathsf{msk}_i, x) \\ & & \mathsf{ct}_f \leftarrow \mathsf{SkEnc}(\{\mathsf{msk}_i\}_{i\in[K]}, f) \\ & & b \leftarrow \mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct}_f) \end{array}\right] > 1 - \mathsf{negl}(\lambda)$$

1-Bounded Function Indistinguishability For any admissible PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{rcl} & & (1^K, 1^\kappa, f^{(0)}, f^{(1)}, U) \leftarrow \mathcal{A}(1^\lambda) \\ & & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa) \\ & & \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ b \leftarrow \mathcal{A}^{\mathsf{SkEnc}(\{\mathsf{msk}_i\}_i, \cdot)}(\mathsf{ct}_b) & : & (\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}, \{\mathsf{msk}_i\}_{i\in U}) \to \mathcal{A}^{\mathsf{SkEnc}(\{\mathsf{msk}_i\}_i, \cdot), \mathsf{KGen}(\cdot, \mathsf{msk}_{(\cdot)}, \cdot)} \\ & & b \leftarrow \{0,1\} \\ & & \mathsf{ct}_b \leftarrow \mathsf{SkEnc}(\{\mathsf{msk}_i\}_{i\in[K]}, f^{(b)}) \end{array}\right] - \frac{1}{2} \leq \mathsf{negl}(\lambda)$$

The SkEnc oracle takes a function $f \in \mathcal{F}_k$ and returns a ciphertext encrypted under f. $\mathcal{A}$ can make only one query to the SkEnc oracle.

The KGen oracle takes authority index $i \in [K]$ and input $x \in \mathcal{X}_\kappa$ and responds with a key generated under i and x.

For any x where $f^{(0)}(x) \neq f^{(1)}(x)$, $\mathcal{A}$ cannot query the KGen oracle for all $\{(i,x)\}_{i\in[K]\setminus U}$, and $U \neq [K]$.

1-Bounded Accept Indistinguishability For any admissible PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{l|ll} & (1^K, 1^\kappa, f, U) \leftarrow \mathcal{A}(1^\lambda) \\ & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa) \\ & \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ b \leftarrow \mathcal{A}^{\mathsf{SkEnc}(\{\mathsf{msk}_i\}_i,\cdot)}(\mathsf{ct}_b) & : & (\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}, \{\mathsf{msk}_i\}_{i\in U}) \rightarrow \mathcal{A}^{\mathsf{SkEnc}(\{\mathsf{msk}_i\}_i,\cdot),\mathsf{KGen}(\cdot,\mathsf{msk}_{(\cdot)},\cdot)} \\ & b \leftarrow \{0,1\} \\ & \mathsf{ct}_0 \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}) \\ & \mathsf{ct}_1 \leftarrow \mathsf{SkEnc}(\{\mathsf{msk}_i\}_{i\in[K]}, f) \end{array}\right] - \frac{1}{2} < \mathsf{negl}(\lambda)$$

The SkEnc oracle takes a function $f \in \mathcal{F}_k$ and returns a ciphertext encrypted under f. $\mathcal{A}$ can make only one query to the SkEnc oracle.

The KGen oracle takes authority index $i \in [K]$ and input $x \in \mathcal{X}_\kappa$ and responds with a key generated under i and x.

For any x where $f(x) = 0$, $\mathcal{A}$ cannot query the KGen oracle for all $\{(i, x)\}_{i\in[K]\setminus U}$, and $U \neq [K]$.

5.3 Multi-Authority PLBE from MA-ABE and DMFE

The idea of this construction is to have a Distributed Mixed-FE Decryption circuit preprogrammed with a DMFE ciphertext as the function policy for the MA-ABE ciphertext. Then the keys will be MA-ABE keys with each attribute being a DMFE secret key. Notably, you must have both *Distributed* Mixed-FE and *Multi-Authority* ABE to create Multi-Authority PLBE. It is not enough to have Distributed Mixed-FE with regular ABE or regular Mixed-FE with Multi-Authority ABE. Let Multi-Authority Attribute-Based Encryption scheme $\mathsf{MABE} = (\mathsf{GSetup}, \mathsf{ASetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ and Distributed Mixed-Functional Encryption scheme $\mathsf{DMFE} = (\mathsf{GSetup}, \mathsf{ASetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{SkEnc})$

$\mathsf{GSetup}(1^\lambda, 1^K, 1^N) \rightarrow \mathsf{crs}$ Here we run the global setup algorithm of both the DMFE scheme and the MA-ABE scheme, with the attribute space of the MA-ABE scheme being the Secret Key space of the MA-Mixed-FE scheme.

$$\mathsf{DMFE}.\mathcal{F}_{\kappa_1} = \{f : [N] \rightarrow \{0,1\}\}$$
$$\mathsf{DMFE.crs} \leftarrow \mathsf{DMFE.GSetup}(1^\lambda, 1^K, 1^{\kappa_1})$$

Let $\mathcal{S}$ be the secret key space of the DMFE scheme.

$$\mathsf{MABE}.\mathcal{F}_{\kappa_2,K} = \{f : \mathcal{S}^K \rightarrow \{0,1\}\}$$
$$\mathsf{MABE.crs} \leftarrow \mathsf{MABE.GSetup}(1^\lambda, 1^{\kappa_2}, 1^K)$$
$$\mathsf{crs} = (\mathsf{MABE.crs}, \mathsf{DMFE.crs})$$

$\mathsf{ASetup}(i) \rightarrow (\mathsf{msk}_i, \mathsf{mpk}_i)$ Run the authority setup algorithms for the DMFE and MA-ABE schemes, then the master public key will be both the master public key of the MA-ABE scheme and the master public key of the DMFE scheme. Then generate all N possible DMFE secret keys for this authority, and generate all N possible ABE secret keys by setting the DMFE secret key as it's attribute. Finally, set the master secret key to the DMFE master secret key along

with all ABE secret keys.

$$(\mathsf{MABE.mpk}_i, \mathsf{MABE.msk}_i) \leftarrow \mathsf{MABE.ASetup}(i)$$
$$(\mathsf{DMFE.mpk}_i, \mathsf{DMFE.msk}_i) \leftarrow \mathsf{DMFE.ASetup}(i)$$
$$\forall j \in [N] : \mathsf{DMFE.sk}_{i,j} \leftarrow \mathsf{DMFE.KGen}(i, \mathsf{DMFE.msk}_i, j)$$
$$\forall j \in [N] : \mathsf{sk}_{i,j} = \mathsf{MABE.sk}_{i,j} \leftarrow \mathsf{MABE.KGen}(i, \mathsf{MABE.msk}_i, \mathsf{DMFE.sk}_{i,j}, j)$$
$$\mathsf{mpk}_i = (\mathsf{MABE.mpk}_i, \mathsf{DMFE.mpk}_i), \mathsf{msk}_i = (\{\mathsf{sk}_{i,j}\}_{j\in[N]}, \mathsf{DMFE.msk}_i)$$

$\mathsf{KGen}(i, \mathsf{msk}_i, j) \to \mathsf{sk}_{i,j}$ Interpret msk_i as the set $(\{\mathsf{sk}_{i,j}\}_{j\in[N]}, \mathsf{DMFE.msk}_i)$ then return $\mathsf{sk}_{i,j}$

$\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m) \to \mathsf{ct}$ Use the MA-ABE encryption algorithm to encrypt the message m with the policy being the decryption circuit for our DMFE scheme, with a preprogrammed public-key mode ciphertext inside.

$$\mathsf{PKDec}(\cdot) = \mathsf{DMFE.Dec}(\ \cdot\ , \mathsf{DMFE.Enc}(\{\mathsf{DMFE.mpk}_i\}))$$
$$\mathsf{ct} \leftarrow \mathsf{MABE.Enc}(\{\mathsf{MABE.mpk}_i\}_{i\in[K]}, \mathsf{PKDec}, m)$$

$\mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, j, m) \to \mathsf{ct}$ Use the MA-ABE encryption algorithm to encrypt the message m with the policy being the decryption circuit for the DMFE scheme with a secret-key mode ciphertext inside. This secret-key ciphertext itself encrypts the function that checks if the given user index is greater than j.

$$\mathsf{comp}_j(\cdot) = (\cdot) \overset{?}{>} j$$
$$\mathsf{SKDec}(\cdot) = \mathsf{DMFE.Dec}(\ \cdot\ , \mathsf{DMFE.SkEnc}(\{\mathsf{DMFE.msk}_i\}_{i\in[K]}, \mathsf{comp}_j))$$
$$\mathsf{ct} \leftarrow \mathsf{MABE.Enc}(\{\mathsf{MABE.mpk}_i\}_{i\in[K]}, \mathsf{SKDec}, m)$$

$\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct}) \to m/\perp$ Since ciphertexts are ABE ciphertexts and secret keys are ABE secret keys, decryption is just ABE decryption.

$$m/\perp \leftarrow \mathsf{MABE.Dec}(\{\mathsf{MABE.sk}_{i,j}\}_{i\in[K]}, \mathsf{ct})$$

Theorem 5.3. If MABE is a valid Multi-Authority Attribute-Based Encryption scheme, and DMFE is a valid Distributed Mixed-Functional Encryption scheme, then the construction on page 19 is a valid Private Linear Broadcast Encryption scheme.

Correctness Over all choices of $\lambda \in \mathbb{N}$, $N \in \mathsf{poly}(\lambda)$, $K \in \mathsf{poly}(\lambda)$, $j \in [N]$, and $m \in M_\lambda$. First evaluate the following algorithms:

$$\mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N)$$
$$\forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i)$$
$$\forall\ i \in [K], \mathsf{sk}_{i,j} \leftarrow \mathsf{KGen}(i, \mathsf{msk}_i, j)$$

Lemma 5.4. If the correctness properties hold for the MA-ABE scheme MABE, and the DMFE scheme DMFE, then with high probability $\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m)) = m$.

Proof.

$$\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m)) =$$
$$\mathsf{MABE.Dec}(\{\mathsf{MABE.sk}_{i,j}\}_{i\in[K]}, \mathsf{MABE.Enc}(\{\mathsf{MABE.mpk}_i\}_{i\in[K]}, \mathsf{PKDec}, m))$$

MA-ABE correctness will only evaluate to m here if the circuit PKDec evaluates to 1. So we first use DMFE correctness to show the following.

$$\mathsf{PKDec}(\{\mathsf{DMFE.sk}_{i,j}\}_{i\in[K]}) =$$
$$\mathsf{DMFE.Dec}(\{\mathsf{DMFE.sk}_{i,j}\}_{i\in[K]}, \mathsf{DMFE.Enc}(\{\mathsf{DMFE.mpk}_i\})) = 1$$

So, we can conclude that $\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m)) = m$. □

Lemma 5.5. If the correctness properties hold for the MA-ABE scheme MABE, and the DMFE scheme DMFE, then with high probability over all choices of $k \in [0, ..., \lambda]$ when $j > k$,

$$\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, k, m)) = m$$

Proof.

$$\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, k, m)) =$$
$$\mathsf{MABE.Dec}(\{\mathsf{MABE.sk}_{i,j}\}_{i\in[K]}, \mathsf{MABE.Enc}(\{\mathsf{MABE.mpk}_i\}_{i\in[K]}, \mathsf{SKDec}, m))$$

MA-ABE correctness will only evaluate to m here if the circuit SKDec evaluates to 1. Notably, MA-ABE correctness tells us that Dec will also evaluate to $\perp$ when the circuit SKDec evaluates to 0. So to prove the lemma above we need to show that SKDec evaluates to 1 if and only if $j > k$. To do this, we use DMFE correctness to show the following.

$$\mathsf{SKDec}(\{\mathsf{DMFE.sk}_{i,j}\}_{i\in[K]}) =$$
$$\mathsf{DMFE.Dec}(\{\mathsf{DMFE.sk}_{i,j}\}_{i\in[K]}, \mathsf{DMFE.SkEnc}(\{\mathsf{DMFE.msk}_i\}_{i\in[K]}, \mathsf{comp}_k)) =$$
$$\mathsf{comp}_k(j) = j \overset{?}{>} k$$

So, we can conclude that if $j > k$.,

$$\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, k, m)) = m$$

□

Normal Hiding Security

Lemma 5.6. If DMFE is a DMFE scheme with the Accept Indistinguishability property then the construction on page 19 satisfies the Normal Hiding Security property.

Proof. Consider a PPT adversary $\mathcal{A}$ on the Normal Hiding Security of our scheme and a DMFE Accept Indistinguishability Challenger $\mathcal{C}$. Then reduction $\mathcal{R}$ is as follows:

$\mathcal{R}$ receives 1^λ from $\mathcal{C}$ and forwards it to $\mathcal{A}$.

Setup Phase $(1^K, 1^N, U)$ $\mathcal{R}$ queries the DMFE challenger $\mathcal{C}$ with $(1^K, 1^{\kappa_1}, f^*(x) = (x \overset{?}{>} 0), U)$ and receives $(\mathsf{DMFE.crs}, \{\mathsf{DMFE.mpk}_i\}_{i\in[K]}, \{\mathsf{DMFE.msk}_i\}_{i\in U})$. Then $\mathcal{R}$ sets up the MABE scheme $\mathsf{MABE.crs} \leftarrow \mathsf{MABE.GSetup}(1^\lambda, 1^K, 1^{\kappa_2})$, and generates all MABE authority keys $\forall\ i \in [K] : (\mathsf{MABE.mpk}_i, \mathsf{MABE.msk}_i) \leftarrow \mathsf{MABE.ASetup}(i)$. Then $\mathcal{R}$ queries $\mathcal{C}$ for all possible DMFE secret keys for the corrupted authorities, so $\forall i \in U, j \in [N]$: Send (i, j) to $\mathcal{C}$, and receives $\{\mathsf{DMFE.sk}_{i,j}\}_{i\in[U],j\in[N]}$. $\mathcal{R}$ turns these secret keys into ABE secret keys $\forall i \in U, j \in [N] : \mathsf{sk}_{i,j} \leftarrow \mathsf{MABE.KGen}(i, \mathsf{MABE.msk}_i, \mathsf{sk}_{i,j}, j)$. Finally $\mathcal{R}$ sends $\mathcal{A}$ the common reference string $\mathsf{crs} = (\mathsf{MABE.crs}, \mathsf{DMFE.crs})$, master public keys $\forall\ i \in [K] : \mathsf{mpk}_i = (\mathsf{MABE.mpk}_i, \mathsf{DMFE.mpk}_i)$, and corrupted master secret keys $\forall\ i \in U : \mathsf{msk}_i = (\{\mathsf{sk}_{i,j}\}_{j\in[N]}, \mathsf{DMFE.msk}_i)$.

1-Bounded Query (m^*) $\mathcal{R}$ uses the 1-Bounded SkEnc encryption query with $\mathcal{C}$ to query f^*, and receives $\mathsf{DMFE.ct}^*$. Then let the circuit $\mathsf{SKDec}^*(\cdot) = \mathsf{DMFE.Dec}(\cdot, \mathsf{DMFE.ct}^*)$. Using the MA-ABE encryption algorithm $\mathcal{R}$ sends $\mathcal{A}$ the ciphertext $\mathsf{ct}^* \leftarrow \mathsf{MABE.Enc}(\{\mathsf{MABE.mpk}_i\}_{i\in[K]}, \mathsf{SKDec}^*, m^*)$.

Key Generation Query (i, j) $\mathcal{R}$ queries the DMFE challenger $\mathcal{C}$ for the key (i, j), and receives $\mathsf{DMFE.sk}_{i,j}$. Then $\mathcal{R}$ sends $\mathcal{A}$ the MA-ABE secret key $\mathsf{MABE.sk}_{i,j} \leftarrow \mathsf{MABE.KGen}(i, \mathsf{msk}_i, \mathsf{DMFE.sk}_{i,j}, j)$

Challenge Phase (m) The DMFE Challenger $\mathcal{C}$ sends $\mathcal{R}$ the challenge ciphertext $\mathsf{DMFE.ct}$. Let the circuit $\mathsf{SKDec}(\cdot) = \mathsf{DMFE.Dec}(\cdot, \mathsf{DMFE.ct})$. Then send $\mathcal{A}$ the challenge ciphertext $\mathsf{ct} \leftarrow \mathsf{MABE.Enc}(\{\mathsf{MABE.mpk}_i\}_{i\in[K]}, \mathsf{SKDec}, m)$. Finally, $\mathcal{A}$ sends b' and $\mathcal{R}$ forwards b' to $\mathcal{C}$.

$\mathcal{R}$ is an admissible adversary against $\mathcal{C}$ since all key generation queries will be for $j > 0$, so $f^*(j) = 1$ for all (i, j) queried. If $\mathcal{A}$ has advantage δ against the Normal Hiding Security of our scheme, then $\mathcal{R}$ will have advantage δ against the DMFE Accept Indistinguishability challenger $\mathcal{C}$.
Having non-negligible advantage is a contradiction, so Normal Hiding Security holds for our scheme. □

Index Hiding Security

Lemma 5.7. If DMFE is a DMFE scheme with the Function Indistinguishability property then the construction on page 19 satisfies the Index Hiding Security property.

Proof. Consider a PPT adversary $\mathcal{A}$ on the Index Hiding Security of our scheme and a DMFE Function Indistinguishability Challenger $\mathcal{C}$. Then reduction $\mathcal{R}$ is as follows:

$\mathcal{R}$ receives 1^λ from $\mathcal{C}$ and forwards it to $\mathcal{A}$.

Setup Phase $(1^K, 1^N, j, U)$ $\mathcal{R}$ queries the DMFE challenger $\mathcal{C}$ with $(1^K, 1^{\kappa_1}, f^{(0)}(\cdot) = (\cdot) \overset{?}{>} j, f^{(1)}(\cdot) = (\cdot) \overset{?}{>} j-1, U)$ and receives $(\mathsf{DMFE.crs}, \{\mathsf{DMFE.mpk}_i\}_{i\in[K]}, \{\mathsf{DMFE.msk}_i\}_{i\in U})$. Then $\mathcal{R}$ sets up the MABE scheme $\mathsf{MABE.crs} \leftarrow \mathsf{MABE.GSetup}(1^\lambda, 1^K, 1^{\kappa_2})$, and generates all MABE authority keys $\forall\ i \in [K] : (\mathsf{MABE.mpk}_i, \mathsf{MABE.msk}_i) \leftarrow \mathsf{MABE.ASetup}(i)$. Then $\mathcal{R}$ queries $\mathcal{C}$ for all possible DMFE secret keys for the corrupted authorities, so $\forall i \in U, j \in [N]$: Send (i, j) to $\mathcal{C}$, and receives $\{\mathsf{DMFE.sk}_{i,j}\}_{i\in[U],j\in[N]}$. $\mathcal{R}$ turns these secret keys into ABE secret keys $\forall i \in U, j \in [N] : \mathsf{sk}_{i,j} \leftarrow \mathsf{MABE.KGen}(i, \mathsf{MABE.msk}_i, \mathsf{sk}_{i,j}, j)$. Finally $\mathcal{R}$ sends $\mathcal{A}$ the common reference string $\mathsf{crs} = (\mathsf{MABE.crs}, \mathsf{DMFE.crs})$, master public keys $\forall\ i \in [K] : \mathsf{mpk}_i = (\mathsf{MABE.mpk}_i, \mathsf{DMFE.mpk}_i)$, and corrupted master secret keys $\forall\ i \in U : \mathsf{msk}_i = (\{\mathsf{sk}_{i,j}\}_{j\in[N]}, \mathsf{DMFE.msk}_i)$.

1-Bounded Query (j^*, m^*) $\mathcal{R}$ uses the 1-Bounded SkEnc encryption query with $\mathcal{C}$ to query $f^*(\cdot) = (\cdot) \overset{?}{>} j^*$, and receives $\mathsf{DMFE.ct}^*$. Then let the circuit $\mathsf{SKDec}^*(\cdot) = \mathsf{DMFE.Dec}(\cdot, \mathsf{DMFE.ct}^*)$. Using the MA-ABE encryption algorithm $\mathcal{R}$ sends $\mathcal{A}$ the ciphertext $\mathsf{ct}^* \leftarrow \mathsf{MABE.Enc}(\{\mathsf{MABE.mpk}_i\}_{i\in[K]}, \mathsf{SKDec}^*, m^*)$.

Key Generation Query (i, j) $\mathcal{R}$ queries the DMFE challenger $\mathcal{C}$ for the key (i, j), and receives $\mathsf{DMFE.sk}_{i,j}$. Then $\mathcal{R}$ sends $\mathcal{A}$ the MA-ABE secret key $\mathsf{MABE.sk}_{i,j} \leftarrow \mathsf{MABE.KGen}(i, \mathsf{msk}_i, \mathsf{DMFE.sk}_{i,j}, j)$

Challenge Phase (m) The DMFE Challenger $\mathcal{C}$ sends $\mathcal{R}$ the challenge ciphertext $\mathsf{DMFE.ct}$. Let the circuit $\mathsf{SKDec}(\cdot) = \mathsf{DMFE.Dec}(\cdot, \mathsf{DMFE.ct})$. Then send $\mathcal{A}$ the challenge ciphertext

$\mathsf{ct} \leftarrow \mathsf{MABE.Enc}(\{\mathsf{MABE.mpk}_i\}_{i\in[K]}, \mathsf{SKDec}, m)$. Finally, $\mathcal{A}$ sends b' and $\mathcal{R}$ forwards b' to $\mathcal{C}$.

If $\mathcal{A}$ is an admissible adversary then it won't query all $\{(i,j)\}_{i\notin U}$. Since j is the only x such that $(x > j) \neq (x > j-1)$ i.e. $f^{(0)}(x) \neq f^{(1)}(x)$ this means $\mathcal{R}$ will be an admissible adversary against $\mathcal{C}$.
If $\mathcal{A}$ has advantage δ against the Index Hiding Security of our scheme, then $\mathcal{R}$ will have advantage δ against the DMFE Function Indistinguishability challenger $\mathcal{C}$.
Having non-negligible advantage is a contradiction, so Index Hiding Security holds for our scheme. □

Message Hiding Security

Lemma 5.8. If MABE is a MA-ABE scheme with the Static Security property then the construction on page 19 satisfies the Message Hiding Security property.

Proof. Consider a PPT adversary $\mathcal{A}$ on the Message Hiding Security of our scheme and a MA-ABE Selective Security Challenger $\mathcal{C}$. Then reduction $\mathcal{R}$ is as follows:

$\mathcal{R}$ receives 1^λ from $\mathcal{C}$ and forwards it to $\mathcal{A}$.

Setup $(1^K, 1^N, U)$ $\mathcal{R}$ sets up the DMFE scheme $\mathsf{DMFE.crs} \leftarrow \mathsf{DMFE.GSetup}(1^\lambda, 1^K, 1^{\kappa_1})$, generates all authority keys $\forall\ i \in [K] : (\mathsf{DMFE.mpk}_i, \mathsf{DMFE.msk}_i) \leftarrow \mathsf{DMFE.ASetup}(i)$, and generates all possible DMFE secret keys $\forall\ i \in [K], j \in [N] : \mathsf{DMFE.sk}_{i,j} \leftarrow \mathsf{DMFE.KGen}(i, \mathsf{DMFE.msk}_i, j)$. Let $Q = \{(i, \mathsf{DMFE.sk}_{i,j}, j)\}_{i\in[K],j\in[N]}$, and let the circuit

$$\mathsf{SKDec}(\cdot) = \mathsf{DMFE.Dec}(\ \cdot\ , \mathsf{DMFE.SkEnc}(\{\mathsf{DMFE.msk}_i\}_{i\in[K]}, \mathsf{comp}_N))$$

Then $\mathcal{R}$ queries MA-ABE challenger $\mathcal{C}$ with $(1^K, 1^{\kappa_2}, \mathsf{SKDec}, Q, \emptyset)$, and receive $(\mathsf{MABE.crs}, \{\mathsf{MABE.mpk}_i\}_{i\in[K]}, \{\mathsf{MABE.sk}_{i,x,\mathsf{gid}}\}_{(i,x,\mathsf{gid})\in Q})$. Finally, $\mathcal{R}$ sends the adversary common reference string $\mathsf{crs} = (\mathsf{MABE.crs}, \mathsf{DMFE.crs})$, master public keys $\{\mathsf{mpk}_i = (\mathsf{MABE.mpk}_i, \mathsf{DMFE.mpk}_i)\}_{i\in[K]}$, and corrupted master secret keys $\{\mathsf{msk}_i = (\{\mathsf{sk}_{i,j}\}_{j\in[N]}, \mathsf{DMFE.msk}_i)\}_{i\in U}$.

1-Bounded Query (j, m^*) $\mathcal{R}$ uses the DMFE secret key encryption algorithm to generate $\mathsf{DMFE.ct}^* \leftarrow \mathsf{SkEnc}(\{\mathsf{msk}_i\}_{i\in[K]}, \mathsf{comp}_j)$. Let the circuit $\mathsf{SKDec}^*(\cdot) = \mathsf{DMFE.Dec}(\ \cdot\ , \mathsf{DMFE.ct}^*)$. Then send $\mathcal{A}$ the ciphertext $\mathsf{ct}^* \leftarrow \mathsf{MABE.Enc}(\{\mathsf{MABE.mpk}_i\}_{i\in[K]}, \mathsf{SKDec}^*, m^*)$.

Key Generation Query (i, j) $\mathcal{R}$ sends $\mathcal{A}$ the MA-ABE secret key $\mathsf{MABE.sk}_{i,\mathsf{DMFE.sk}_{i,j},j}$.

Challenge Phase (m_0, m_1) $\mathcal{R}$ forwards (m_0, m_1) to $\mathcal{C}$ and receives the MA-ABE challenge ciphertext MABE.ct. $\mathcal{R}$ forwards MABE.ct to $\mathcal{A}$ as the MA-PLBE ciphertext. Lastly, $\mathcal{A}$ sends b' and $\mathcal{R}$ forwards b' to $\mathcal{C}$.

By correctness of DMFE scheme we know that $\mathsf{SKDec}(\{\mathsf{DMFE.sk}_{i,j}\}_{i\in[K]}) = 0$ with high probability, since as long as $\mathcal{A}$ is an admissible adversary we know for all queried $j : j \leq N$ so $\mathsf{comp}_N(j) = 0$. We also *don't* use any ABE authority corruptions, so we don't have to worry about corrupted ABE secret keys. This means $\mathcal{R}$ is an admissible adversary against $\mathcal{C}$. If $\mathcal{A}$ has advantage δ against the Message Hiding Security of our scheme, then $\mathcal{R}$ will have advantage δ against the MA-ABE Selective Security challenger $\mathcal{C}$.
Having non-negligible advantage is a contradiction, so Message Hiding Security holds for our scheme. □

6 Distributed Mixed Functional Encryption from LWE

6.1 Preliminary: Almost Key-Homomorphic Private-Constrained PRF

Key-Homomorphic Private-Constrained PRFs are first described and constructed in [CVW+18a]. They use Private Constrained PRF constructions in [CC17], [CVW18b], and [BTVW17] combined with key-homomorphism to create the object.

Syntax. An Almost Key-Homomorphic Private Constrained PRF (PCPRF) is a family of functions $\mathcal{F} = \{\mathcal{F}_{\mathcal{K}_\lambda} : \mathcal{D}_\kappa \to \mathcal{R}_\kappa\}_{\kappa\in\mathbb{N}}$, with corresponding functional constraint family $\mathcal{C} = \{\mathcal{C}_\kappa : \mathcal{D}_\kappa \to \{0,1\}\}_\kappa$, group operation $\circ$ over R_κ and $\mathcal{K}_\kappa$, and the following polynomial time algorithms.

$\mathsf{ppGen}(1^\lambda, 1^\kappa) \to \mathsf{crs}$ The randomized public key generation algorithm generates common reference string crs using a security parameter λ and a function parameter κ. crs is assumed as input to the following functions.

$\mathsf{skGen}(\mathsf{crs}) \to \mathsf{sk}$ The randomized secret key algorithm uses common reference string crs to generate a secret key $\mathsf{sk} \in \mathcal{K}_\kappa$ which is an index for the function class $\mathcal{F}$.

$\mathsf{Eval}(\mathsf{sk}, x) \to y$ The deterministic evaluation algorithm that simply evaluates the function $f_{\mathsf{sk}} \in \mathcal{F}_\kappa$ on the input $x \in \mathcal{D}_\kappa$.

$\mathsf{Constrain}(\mathsf{sk}, C) \to \mathsf{ck}_C$ The randomized constrain algorithm which generates a constrained key ck_C which is an index for the function class $\mathcal{F}_\kappa$. Takes as input the common reference string crs, secret key sk, and constraint function $C \in \mathcal{C}_\kappa$.

$\mathsf{CEval}(\mathsf{ck}_C, x) \to y$ The deterministic constrained evaluation algorithm that simply evaluates the input $x \in \mathcal{D}_\kappa$ on the constrained key ck_C.

Definition 6.1. An PCPRF family of functions $\mathcal{F}$ with (ppGen, skGen, Eval, $\mathsf{Constrain}$, CEval) must satisfy the following properties.

Functionality Preservation There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$, $\kappa \in \mathsf{poly}(\lambda)$, $x \in \mathcal{D}_\kappa$, and $C \in \mathcal{C}_\kappa$ where $C(x) = 0$[4],

$$\Pr\left[\mathsf{Eval}(\mathsf{sk}, x) = \mathsf{CEval}(\mathsf{ck}_C, x) \;:\; \begin{array}{l}\mathsf{crs} \leftarrow \mathsf{ppGen}(1^\lambda, 1^\kappa)\\ \mathsf{sk} \leftarrow \mathsf{skGen}(\mathsf{crs})\\ \mathsf{ck}_C \leftarrow \mathsf{Constrain}(\mathsf{sk}, C)\end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Pseudorandomness and Constraint-Hiding There exists a polynomial time algorithm SIM such that for every stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\mathcal{A}^{\mathsf{Eval}(\mathsf{sk},\cdot)}(\mathsf{crs}, \mathsf{ck}_C) = 1 \;:\; \begin{array}{l}C \leftarrow \mathcal{A}(1^\lambda)\\ \mathsf{crs} \leftarrow \mathsf{ppGen}(1^\lambda, 1^{|C|})\\ \mathsf{sk} \leftarrow \mathsf{skGen}(\mathsf{crs})\\ \mathsf{ck}_C \leftarrow \mathsf{Constrain}(\mathsf{sk}, C)\end{array}\right] -$$

$$\Pr\left[\mathcal{A}^{\mathsf{SIM.Eval}(\cdot)}(\mathsf{crs}, \mathsf{ck}_C) = 1 \;:\; \begin{array}{l}C \leftarrow \mathcal{A}(1^\lambda)\\ (\mathsf{crs}, \mathsf{ck}_C) \leftarrow \mathsf{SIM}(1^\lambda, 1^{|C|})\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

[4]By lattice convention $C(x) = 0$ is our accept condition and $C(x) = 1$ is our reject condition.

The $\mathsf{SIM.Eval}$ oracle responds with $y = \mathsf{CEval}(\mathsf{ck}_C, x)$ when $C(x) = 0$ and responds with $y \leftarrow U(\mathcal{R}_\kappa)$ when $C(x) = 1$.

Almost Key Homomorphism Over all $\lambda \in \mathbb{N}$, $\kappa \in \mathsf{poly}(\lambda)$, $x \in \mathcal{D}_\kappa$, and $\mathsf{sk}_1, \mathsf{sk}_2 \in \mathcal{K}_\kappa$ the following holds.

$$||\mathsf{Eval}(\mathsf{sk}_1 \circ \mathsf{sk}_2, x) - \mathsf{Eval}(\mathsf{sk}_1, x) \circ \mathsf{Eval}(\mathsf{sk}_2, x)||_\infty \leq 1$$

Distribution Requirement Over all $\lambda \in \mathbb{N}$, $\kappa \in \mathsf{poly}(\lambda)$, and $\mathsf{sk}^* \in \mathcal{K}_\lambda$. First get $\mathsf{crs} \leftarrow \mathsf{ppGen}(1^\lambda, 1^\kappa)$, then sample $\mathsf{sk}_1 \leftarrow \mathsf{skGen}(\mathsf{crs}), \mathsf{sk}_2 \leftarrow \mathsf{skGen}(\mathsf{crs})$. We want the following distribution requirement to hold over the randomness of skGen.

$$\Pr\left[\mathsf{sk}_1 = \mathsf{sk}^*\right] = \Pr\left[\mathsf{sk}_1 \circ \mathsf{sk}_2 = \mathsf{sk}^*\right] = \Pr\left[\mathsf{sk}_1 \circ (-\mathsf{sk}_2) = \mathsf{sk}^*\right]$$

Theorem 6.2. Assuming the hardness of LWE, there exists an Almost Key-Homomorphic Private-Constrained PRF satisfying all of the above properties.

Proof Sketch. The construction for Almost Key-Homomorphic Private-Constrained PRF's was first described in Section 4.1 of [CVW+18a], they use the Private-Constrained PRF's from [CC17] and [CVW18b]. For a more detailed construction of KHPCPRF's see Section 6.1 of [GQWW19]. This construction gives us a KHPCPRF for comparison functions with constrained key size $O(|D_\kappa|^\epsilon)$ $\mathsf{poly}(\lambda)$ for some $\epsilon > 0$, and constrained evaluation in $\mathsf{NC1}$. □

6.2 DMFE from PCPRF

The main idea behind this construction is to take the construction of 1-Bounded Mixed-FE from [CVW+18a] and decentralize it. We have each authority create their own set of secret keys which they can distribute to users separately. Then, encryption combines all authorities' secret keys into one key using key homomorphism, and constrains that combined key.

$\mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa) \rightarrow \mathsf{crs}$ Set up the PCPRF by running $\mathsf{PCPRF.crs} \leftarrow \mathsf{ppGen}(1^\lambda, 1^\kappa)$ and then combine the PCPRF common reference string with the number of authorities to create the DMFE common reference string $\mathsf{crs} = (\mathsf{PCPRF.crs}, 1^K)$.

$\mathsf{ASetup}(i) \rightarrow (\mathsf{mpk}_i, \mathsf{msk}_i)$ Create 2λ many secret keys $\forall\ j \in [\lambda], b \in \{0,1\}$ $\mathsf{sk}_{i,j,b} \leftarrow \mathsf{skGen}(\mathsf{crs})$. Let the DMFE master secret key be exactly this set of generated secret keys $\mathsf{msk}_i = \{\mathsf{sk}_{i,j,b}\}_{j\in[\lambda],b\in\{0,1\}}$.

$\mathsf{KGen}(i, \mathsf{msk}_i, x) \rightarrow \mathsf{sk}_{i,x}$ Key generation will output the evaluation of all the authorities' secret keys on $x \in D_\kappa$, $\mathsf{sk}_{i,x} = \{\mathsf{Eval}(\mathsf{sk}_{i,j,b}, x)\}_{j\in[\lambda],b\in\{0,1\}}$.

$\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}) \rightarrow \mathsf{ct}$ Simply run the PCPRF's simulator algorithm SIM to generate a fake constrained key $(\mathsf{crs}, \mathsf{ck}_1) \leftarrow \mathsf{SIM}(1^\lambda, 1^\kappa)$. Randomly sample a vector $r \leftarrow \{0,1\}^\lambda$. Then let the DMFE ciphertext be the random vector r along with the simulated constrained key $\mathsf{ct} = (r, \mathsf{ck}_C)$.

$\mathsf{SkEnc}(\{\mathsf{msk}_i\}_{i\in[K]}, f) \rightarrow \mathsf{ct}$ Here we first sample a random vector $r \leftarrow \{0,1\}^\lambda$. Then use r to create a unique secret key $\mathsf{sk}'_i = \sum_{j\in[\lambda]} \mathsf{sk}_{i,j,r_j}$ for each authority. Next we add together all of these secret keys and constrain $\mathsf{ck}_f \leftarrow \mathsf{Constrain}\left(\sum_{i=1}^K \mathsf{sk}'_i, f\right)$. Finally let the DMFE ciphertext just be r alongside this constrained key $\mathsf{ct} = (r, \mathsf{ck}_f)$.

$\mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct}) \to \{0,1\}$ Interpret ct as (r, ck), and reconstruct the secret key evaluation at x using all of the secret keys by adding up $\mathsf{Eval}(\mathsf{sk}_{i,j,r_j}, x)$ for all $i \in [K]$ and $j \in [\lambda]$. Compare this sum to the constrained evaluation of x on the constrained key given in the ciphertext.

$$\left|\left|\left(\sum_{i\in[K]}\sum_{j\in[\lambda]} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j}, x)\right) - \mathsf{CEval}(\mathsf{ck}, x)\right|\right|_\infty \leq K\lambda : \text{output } 0$$

$$\left|\left|\left(\sum_{i\in[K]}\sum_{j\in[\lambda]} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j}, x)\right) - \mathsf{CEval}(\mathsf{ck}, x)\right|\right|_\infty > K\lambda : \text{output } 1$$

Theorem 6.3. If PCPRF is a valid Almost Key-Homomorphic Private-Constrained PRF, then the construction on page 25 is a valid Distributed Mixed-FE scheme.

Correctness For all $\lambda \in \mathbb{N}$, $K \in \mathsf{poly}(\lambda)$, $\kappa \in \mathsf{poly}(\lambda)$, $x \in D_\lambda$, $f \in \mathcal{F}_\kappa$ evaluate the following algorithms.

$$\mathsf{crs} \leftarrow \mathsf{DMFE.GSetup}(1^\lambda, 1^K, 1^\kappa)$$
$$\forall\, i \in [K] : (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{DMFE.ASetup}(i)$$
$$\forall\, i \in [K] : \mathsf{sk}_{i,x} \leftarrow \mathsf{KGen}(i, \mathsf{msk}_i, x)$$
$$\mathsf{ct}_1 \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]})$$
$$\mathsf{ct}_f \leftarrow \mathsf{SkEnc}(\{\mathsf{msk}_i\}_{i\in[K]}, f)$$

Lemma 6.4. Taken over the randomness of PCPRF's ppGen, skGen, and SIM algorithms

$$\Pr\left[\mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct}_1) = 1\right] \geq 1 - \mathsf{negl}(\lambda)$$

.

Proof. Interpret ct_1 as (r_1, ck_1), and interpret $\mathsf{sk}_{i,x}$ as $\{\mathsf{Eval}(\mathsf{sk}_{i,j,b}, x)\}_{j\in[\lambda],b\in\{0,1\}}$. Let $y' = \sum_{i\in[K]} \sum_{j\in[\lambda]} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j}, x)$, then the Dec algorithm above will output 1 in the event $|y' - \mathsf{CEval}(\mathsf{ck}_1, x)| > K\lambda$. So, consider the following probability over the randomness of SIM when generating ck_1 and the randomness of ppGen and skGen when generating the secret keys used to calculate y'.

$$\Pr\left[||y' - y||_\infty > K\lambda \mid y = \mathsf{CEval}(\mathsf{ck}_1, x)\right]$$

Since ck_1 is a simulated constrained key it's evaluations must be pseudorandom.

$$\Pr\left[||y' - y||_\infty > K\lambda \mid y \leftarrow U(\mathcal{R}_\kappa)\right]$$

If we chose as sufficiently large output space $|\mathcal{R}_\kappa| \geq (2K\lambda + 1)2^{\mathsf{polylog}(\lambda)}$ then we can bound the probability of the event $||y' - y||_\infty$.

$$\Pr\left[||y' - y||_\infty > K\lambda \mid y \leftarrow U(\mathcal{R}_\kappa)\right] \geq 1 - \frac{2K\lambda + 1}{(2K\lambda + 1)2^{\mathsf{polylog}(\lambda)}} = 1 - \mathsf{negl}(\lambda)$$

Going back to the Dec algorithm's description, we get $\mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct}_1) = 1$ if and only if $||y' = y||_\infty \geq K\lambda$. So,

$$\Pr\left[\mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct}_1) = 1\right] \geq 1 - \mathsf{negl}(\lambda)$$

□

Lemma 6.5. Taken over the randomness of PCPRF's ppGen, skGen, and Constrain algorithms. $\Pr\left[\mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct}_f) = f(x)\right] > 1 - \mathsf{negl}(\lambda)$

Proof. Interpret ct_f as (r, ck_f), interpret ck_f as $\mathsf{Constrain}(\mathsf{sk}^*, f)$ where

$$\mathsf{sk}^* = \sum_{i\in[K]} \sum_{j\in[\lambda]} \mathsf{sk}_{i,j,r_j}$$

$$\text{Let } y' = \sum_{j\in[\lambda]} \sum_{i\in[K]} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j}, x)$$

Using the Almost Key Homomorphism property of our PCPRF and the triangle inequality we get:

$$||\mathsf{Eval}(\mathsf{sk}_{i,j,r_j}, x) + \mathsf{Eval}(\mathsf{sk}_{i+1,j,r_j}, x) - \mathsf{Eval}(\mathsf{sk}_{i,j,r_j} + \mathsf{sk}_{i+1,j,r_j}, x))||_\infty \leq 1$$

$$||\sum_{i\in[K]} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j}, x) - \mathsf{Eval}(\sum_{i\in[K]} \mathsf{sk}_{i,j,r_j})||_\infty \leq K$$

$$||\sum_{j\in[\lambda]} \sum_{i\in[K]} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j}, x) - \mathsf{Eval}(\sum_{j\in[\lambda]} \sum_{i\in[K]} \mathsf{sk}_{i,j,r_j}, x)||_\infty \leq K\lambda$$

$$||y' - \mathsf{Eval}(\mathsf{sk}^*, x)||_\infty \leq K\lambda$$

If $f(x) = 1$ then using the Pseudorandomness and Constraint Hiding property $\mathsf{Eval}(\mathsf{sk}^*, x)$ must look pseudorandom with respect to the constrained evaluation $\mathsf{CEval}(\mathsf{ck}_f, x)$. For the sake of contradiction, suppose this was not the case, then we would be able to distinguish the $\mathsf{SIM.Eval}$ oracle response of $y \leftarrow U(\mathcal{R}_\kappa)$ from the Eval oracle response $y = \mathsf{Eval}(\mathsf{sk}^*, x)$. So using the same argument as that for the public-key mode, for a large enough $\mathcal{R}_\kappa$ we get.

$$\Pr\left[|\mathsf{Eval}(\mathsf{sk}^*, x) - \mathsf{CEval}(\mathsf{ck}_f, x)| > 2K\lambda\right] \geq 1 - \mathsf{negl}(\lambda)$$

$$\Pr\left[|y' - \mathsf{CEval}(\mathsf{ck}_f, x)| > K\lambda\right] \geq 1 - \mathsf{negl}(\lambda)$$

$$\Pr\left[\mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct}_f) = 1\right] \geq 1 - \mathsf{negl}(\lambda)$$

If $f(x) = 0$ then we can use Functionality Preservation to get:

$$\Pr\left[\mathsf{Eval}(\mathsf{sk}^*, x) = \mathsf{CEval}(\mathsf{ck}_f, x)\right] \geq 1 - \mathsf{negl}(\lambda)$$

$$\Pr\left[||y' - \mathsf{CEval}(\mathsf{ck}_f, x)||_\infty \leq K\lambda\right] \geq 1 - \mathsf{negl}(\lambda)$$

$$\Pr\left[\mathsf{Dec}(\{\mathsf{sk}_{i,x}\}_{i\in[K]}, \mathsf{ct}_f) = 0\right] \geq 1 - \mathsf{negl}(\lambda)$$

□

1-Bounded Function Indistinguishability We use the following hybrids to prove the Function Indistinguishability of our Mixed-FE construction.

$\mathsf{Hybrid}^{\mathcal{A}}(0, b)$ The real security game with random bit b.

Setup $(1^K, 1^\kappa, f^{(0)}, f^{(1)}, U)$ The challenger samples $\mathsf{crs} \leftarrow \mathsf{ppGen}(1^\lambda, 1^{|f^{(b)}|})$. Then the challenger samples $r^{(0)}, r^{(1)} \leftarrow \{0,1\}^\lambda$, stopping if $r^{(0)} = r^{(1)}$. Finally, the challenger generates $2K\lambda$ many secret keys $\forall\ i \in [K],\ j \in [\lambda],\ b \in \{0,1\} : \mathsf{sk}_{i,j,b} \leftarrow \mathsf{skGen}(\mathsf{crs})$, and sends $(\mathsf{crs}, \{\mathsf{sk}_{i,j,b}\}_{i\in U, j\in[\lambda], b\in\{0,1\}})$ to $\mathcal{A}$. At this point the challenger can also get $\mathsf{ck}_{f^{(b)}} \leftarrow \mathsf{Constrain}(\sum_{i\in[K], j\in[\lambda]} sk_{i,j,r_j^{(0)}}, f^{(b)})$.

Query $\mathsf{SkEnc}(f^*)$ The challenger calculates $\mathsf{sk}_1 = \sum_{i\in[K]} \sum_{j\in[\lambda]} \mathsf{sk}_{i,j,r_j^{(1)}}$ and $\mathsf{ck}_1 \leftarrow \mathsf{Constrain}(\mathsf{sk}_1, f^*)$. Then responds with $\mathsf{ct}_1 = (r^{(1)}, \mathsf{ck}_1)$.

Query $\mathsf{KGen}(i, x)$ Respond with $\{\mathsf{Eval}(\mathsf{sk}_{i,j,b}, x)\}_{j\in[\lambda], b\in\{0,1\}}$.

Challenge Ciphertext Send $\mathsf{ct} = (r^{(0)}, \mathsf{ck}_{f^{(b)}})$.

$\mathsf{Hybrid}^{\mathcal{A}}(1,b)$ Sample the challenge constrained key during setup instead of during the challenge phase. Then, during key generation we reverse engineer the secret keys to retain the additive distribution we are looking for.

Setup $(1^K, 1^\kappa, f^{(0)}, f^{(1)}, U)$ The challenger samples $\mathsf{crs} \leftarrow \mathsf{ppGen}(1^\lambda, 1^{|f^{(b)}|})$, $\mathsf{sk} \leftarrow \mathsf{skGen}(\mathsf{crs})$ and $\mathsf{ck}_{f^{(b)}} \leftarrow \mathsf{Constrain}(\mathsf{sk}, f^{(b)})$. Then the challenger samples $r^{(0)}, r^{(1)} \leftarrow \{0,1\}^\lambda$ and finds a $k \in [\lambda]$ such that $r_k^{(0)} \neq r_k^{(1)}$. The challenger chooses any honest authority index $h \in [K] \backslash U$. Finally the challenger generates $2K\lambda$ many secret keys $\forall\ i \in [K],\ j \in [\lambda],\ b \in \{0,1\} : \mathsf{sk}_{i,j,b} \leftarrow \mathsf{skGen}(\mathsf{crs})$, and sends $(\mathsf{crs}, \{\mathsf{sk}_{i,j,b}\}_{i \in U, j \in [\lambda], b \in \{0,1\}})$ to $\mathcal{A}$.

Query $\mathsf{SkEnc}(f^*)$ The challenger calculates $\mathsf{sk}_1 = \sum_{i \in [K]} \sum_{j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(1)}}$ and $\mathsf{ck}_1 \leftarrow \mathsf{Constrain}(\mathsf{sk}_1, f^*)$. Then responds with $\mathsf{ct}_1 = (r^{(1)}, \mathsf{ck}_1)$.

Query $\mathsf{KGen}(i,x)$ Respond with $\{\mathsf{Eval}(\mathsf{sk}_{i,j,b}, x)\}_{j \in [\lambda], b \in \{0,1\}}$, if authority h is queried then we replace $\mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ in our response, instead, we put $E_{h,k,x} = \mathsf{Eval}(\mathsf{sk}, x) - \mathsf{Eval}(\sum_{i \in [K], j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(0)}}) + \mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$

Challenge Ciphertext Send $\mathsf{ct} = (r^{(0)}, \mathsf{ck}_{f^{(b)}})$.

$\mathsf{Hybrid}^{\mathcal{A}}(2,b)$ On key generation queries where $f^{(b)}(x) = 1$ we respond normally, on queries where $f^{(b)}(x) = 0$ we use the constrained key to instead to evaluate the secret key query.

Setup $(1^K, 1^\kappa, f^{(0)}, f^{(1)}, U)$ The challenger samples $\mathsf{crs} \leftarrow \mathsf{ppGen}(1^\lambda, 1^{|f^{(b)}|})$, $\mathsf{sk} \leftarrow \mathsf{skGen}(\mathsf{crs})$ and $\mathsf{ck}_{f^{(b)}} \leftarrow \mathsf{Constrain}(\mathsf{sk}, f^{(b)})$. Then the challenger samples $r^{(0)}, r^{(1)} \leftarrow \{0,1\}^\lambda$ and finds a $k \in [\lambda]$ such that $r_k^{(0)} \neq r_k^{(1)}$. The challenger chooses any honest authority index $h \in [K] \backslash U$. Finally the challenger generates $2K\lambda$ many secret keys $\forall\ i \in [K],\ j \in [\lambda],\ b \in \{0,1\} : \mathsf{sk}_{i,j,b} \leftarrow \mathsf{skGen}(\mathsf{crs})$, and sends $(\mathsf{crs}, \{\mathsf{sk}_{i,j,b}\}_{i \in U, j \in [\lambda], b \in \{0,1\}})$ to $\mathcal{A}$.

Query $\mathsf{SkEnc}(f^*)$ The challenger calculates $\mathsf{sk}_1 = \sum_{i \in [K]} \sum_{j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(1)}}$ and $\mathsf{ck}_1 \leftarrow \mathsf{Constrain}(\mathsf{sk}_1, f^*)$. Then responds with $\mathsf{ct}_1 = (r^{(1)}, \mathsf{ck}_1)$.

Query $\mathsf{KGen}(i,x)$ Respond with $\{\mathsf{Eval}(\mathsf{sk}_{i,j,b}, x)\}_{j \in [\lambda], b \in \{0,1\}}$, if authority h is queried then we replace $\mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ in our response, instead, we put $E_{h,k,x} = \mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x) - \mathsf{Eval}(\sum_{i \in [K], j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(0)}}) + \mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ when $f^{(b)}(x) = 0$ and $E_{h,k,x} = \mathsf{Eval}(\mathsf{sk}, x) - \mathsf{Eval}(\sum_{i \in [K], j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(0)}}) + \mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ when $f^{(b)}(x) = 1$.

Challenge Ciphertext Send $\mathsf{ct} = (r^{(0)}, \mathsf{ck}_{f^{(b)}})$.

$\mathsf{Hybrid}^{\mathcal{A}}(3,b)$ We replace the constrained key generated during setup with a simulated constrained key. This, in turn, makes key generation queries pseudorandom.

Setup $(1^K, 1^\kappa, f^{(0)}, f^{(1)}, U)$ The challenger samples $\mathsf{ck}_{f^{(b)}} \leftarrow \mathsf{SIM}.(1^\lambda, 1^{|f^{(b)}|})$. Then the challenger samples $r^{(0)}, r^{(1)} \leftarrow \{0,1\}^\lambda$ and finds a $k \in [\lambda]$ such that $r_k^{(0)} \neq r_k^{(1)}$. The challenger chooses any honest authority index $h \in [K] \backslash U$. Finally the challenger generates $2K\lambda$ many secret keys $\forall\ i \in [K],\ j \in [\lambda],\ b \in \{0,1\} : \mathsf{sk}_{i,j,b} \leftarrow \mathsf{skGen}(\mathsf{crs})$, and sends $(\mathsf{crs}, \{\mathsf{sk}_{i,j,b}\}_{i \in U, j \in [\lambda], b \in \{0,1\}})$ to $\mathcal{A}$.

Query $\mathsf{SkEnc}(f^*)$ The challenger calculates $\mathsf{sk}_1 = \sum_{i \in [K]} \sum_{j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(1)}}$ and $\mathsf{ck}_1 \leftarrow \mathsf{Constrain}(\mathsf{sk}_1, f^*)$. Then responds with $\mathsf{ct}_1 = (r^{(1)}, \mathsf{ck}_1)$.

Query $\mathsf{KGen}(i,x)$ Respond with $\{\mathsf{Eval}(\mathsf{sk}_{i,j,b}, x)\}_{j \in [\lambda], b \in \{0,1\}}$, if authority h is queried then we replace $\mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ in our response, instead, we put $E_{h,k,x} = \mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x) - \mathsf{Eval}(\sum_{i \in [K], j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(0)}}) + \mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ when $f^{(b)}(x) = 0$ and $E_{h,k,x} \leftarrow U(\mathcal{R}_\lambda)$ when $f^{(b)}(x) = 1$.

Challenge Ciphertext Send $\mathsf{ct} = (r^{(0)}, \mathsf{ck}_{f^{(b)}})$.

Lemma 6.6. Assuming the Functionality Preservation and Almost Key-Homomorphism properties of our PCPRF scheme, $\mathsf{Hybrid}^{\mathcal{A}}(0,b)$ and $\mathsf{Hybrid}^{\mathcal{A}}(1,b)$ are statistically indistinguishable.

Proof. The only difference between these two hybrids are how they treat key generation queries for $\mathsf{sk}_{h,k,r_k^{(0)}}$ on some x. So we want to show that with high probability the distribution of possible values of $E_{h,k,x}$ is indistinguishable from the distribution of $\mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ given a constrained key evaluation $y = \mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x)$ and a set of evaluations $\{\mathsf{Eval}(\mathsf{sk}_{i,j,r_j^{(0)}}, x)\}_{i\in[K],j\in[\lambda],(i,j)\neq(h,k)}$. Using the basic distribution requirement it is clear that $E_{h,k,x}$ must have the same distribution as $\mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ when considered on their own. The only distinguishing property between these two comes from the Almost Key Homomorphism property. In $\mathsf{Hybrid}^{\mathcal{A}}(0,b)$ we expect that random variable $y' = \sum_{i\in[K],j\in[\lambda],(i,j)\neq(h,k)} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j^{(0)}}, x) + \mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ to have some distribution centered around y. In particular, using Almost Key-Homomorphism and assuming the worst case where $f^{(b)}(x) = 0$ we know that $||y - y'||_\infty = ||\mathsf{Eval}(\sum_{i\in[K],j\in[\lambda]} \mathsf{sk}_{i,j,r_j^{(0)}}) - \sum_{i\in[K],j\in[\lambda]} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j^{(0)}}, x)||_\infty \leq K\lambda$. In $\mathsf{Hybrid}^{\mathcal{A}}(1,b)$ we create an identical distribution centered around $z = \mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x)$, since if we let

$$z' = \sum_{i\in[K],j\in[\lambda],(i,j)\neq(h,k)} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j^{(0)}}, x) + E_{h,k,x}$$

$$||z - z'||_\infty = ||\mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x) - \sum_{i\in[K],j\in[\lambda],(i,j)\neq(h,k)} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j^{(0)}}, x) - E_{h,k,x}||_\infty$$

$$||z - z'||_\infty = ||\mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x) - \sum_{i\in[K],j\in[\lambda],(i,j)\neq(h,k)} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j^{(0)}}, x) -$$
$$\mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x) + \mathsf{Eval}(\sum_{i\in[K],j\in[\lambda],(i,j)\neq(h,k)} \mathsf{sk}_{i,j,r_j^{(0)}}) - \mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)||_\infty$$

$$||z - z'||_\infty = ||\mathsf{Eval}(\sum_{i\in[K],j\in[\lambda]} \mathsf{sk}_{i,j,r_j^{(0)}}) - \sum_{i\in[K],j\in[\lambda]} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j^{(0)}}, x)||_\infty$$

□

Lemma 6.7. Assuming the Functionality Preservation property of our PCPRF scheme, $\mathsf{Hybrid}^{\mathcal{A}}(1,b)$ and $\mathsf{Hybrid}^{\mathcal{A}}(2,b)$ are statistically indistinguishable.

Proof. This is simply an application of the Functionality Preservation property. □

Lemma 6.8. Assuming the Pseudorandomness and Constraint-Hiding property of our PCPRF scheme, $\mathsf{Hybrid}^{\mathcal{A}}(2,b)$ and $\mathsf{Hybrid}^{\mathcal{A}}(3,b)$ are computationally indistinguishable.

Proof. If an attacker $\mathcal{A}$ is able to distinguish between $\mathsf{Hybrid}^{\mathcal{A}}(2,b)$ and $\mathsf{Hybrid}^{\mathcal{A}}(3,b)$ then they would have to distinguish between the real and simulated constrained keys in the Pseudorandomness and Constraint-Hiding game. □

Lemma 6.9. Assuming $\mathcal{A}$ is a valid Function Indistinguishability attacker then $\mathsf{Hybrid}^{\mathcal{A}}(2,0)$ and $\mathsf{Hybrid}^{\mathcal{A}}(2,1)$ are statistically indistinguishable.

Proof. From an information theoretic perspective the only place any information about $f^{(b)}$ is used by the challenger is in the KGen Query phase. In this phase some information could be leaked if $\mathcal{A}$ queried an x such that $f^{(0)}(x) \neq^{(1)} (x)$, since it would result in slightly different behavior from the challenger. However, an admissible adversary in the Function Indistinguishability game cannot query all $\{(i,x)\}_{i\in[K]\setminus U}$ such that $f^{(0)}(x) \neq f^{(1)}(x)$. So $\mathcal{A}$ could never compile a complete set of secret key evaluations $\{\mathsf{Eval}(\mathsf{sk}_{i,j,r_j^{(0)}}, x)\}_{i\in[K],j\in[\lambda]}$ with this x, meaning $\mathcal{A}$ would not have enough information to calculate the value $\sum_{i\in[K],j\in[\lambda]} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j^{(0)}}, x)$. Since $\mathcal{A}$ lacks full information on this particular x, it cannot compare it to the constrained key evaluation on x. So, it is impossible for $\mathcal{A}$ to learn any information about $f^{(b)}(x)$. So, in turn, $\mathcal{A}$ could not learn any information about b. □

To conclude the Function Indistinguishability proof, we can follow the hybrid indistinguishability from $\mathsf{Hybrid}^{\mathcal{A}}(0,0) \to \mathsf{Hybrid}^{\mathcal{A}}(1,0) \to \mathsf{Hybrid}^{\mathcal{A}}(2,0) \to \mathsf{Hybrid}^{\mathcal{A}}(3,0) \to \mathsf{Hybrid}^{\mathcal{A}}(3,1) \to \mathsf{Hybrid}^{\mathcal{A}}(2,1) \to \mathsf{Hybrid}^{\mathcal{A}}(1,1) \to \mathsf{Hybrid}^{\mathcal{A}}(0,1)$. So the real game with $b = 0$ ($\mathsf{Hybrid}^{\mathcal{A}}(0,0)$) must be indistinguishable from the real game with $b = 1$ ($\mathsf{Hybrid}^{\mathcal{A}}(0,1)$).

1-Bounded Accept Indistinguishability We use the following hybrids to prove the Accept Indistinguishability of our Mixed-FE construction.

$\mathsf{Hybrid}^{\mathcal{A}}(0)$ The real security game in the SkEnc mode.

Setup $(1^K, 1^\kappa, f, U)$ The challenger samples $\mathsf{crs} \leftarrow \mathsf{ppGen}(1^\lambda, 1^{|f|})$. Then the challenger samples $r^{(0)}, r^{(1)} \leftarrow \{0,1\}^\lambda$, stopping if $r^{(0)} = r^{(1)}$. Finally, the challenger generates $2K\lambda$ many secret keys $\forall\ i \in [K],\ j \in [\lambda],\ b \in \{0,1\} : \mathsf{sk}_{i,j,b} \leftarrow \mathsf{skGen}(\mathsf{crs})$, and sends $(\mathsf{crs}, \{\mathsf{sk}_{i,j,b}\}_{i\in U, j\in[\lambda], b\in\{0,1\}})$ to $\mathcal{A}$. At this point the challenger can also get $\mathsf{ck}_f \leftarrow \mathsf{Constrain}(\sum_{i\in[K],j\in[\lambda]} sk_{i,j,r_j^{(0)}}, f^{(b)})$.

Query $\mathsf{SkEnc}(f^*)$ The challenger calculates $\mathsf{sk}_1 = \sum_{i\in[K]} \sum_{j\in[\lambda]} \mathsf{sk}_{i,j,r_j^{(1)}}$ and $\mathsf{ck}_1 \leftarrow \mathsf{Constrain}(\mathsf{sk}_1, f^*)$. Then responds with $\mathsf{ct}_1 = (r^{(1)}, \mathsf{ck}_1)$.

Query $\mathsf{KGen}(i,x)$ Respond with $\{\mathsf{Eval}(\mathsf{sk}_{i,j,b}, x)\}_{j\in[\lambda],b\in\{0,1\}}$.

Challenge Ciphertext Send $\mathsf{ct} = (r^{(0)}, \mathsf{ck}_f)$.

$\mathsf{Hybrid}^{\mathcal{A}}(1)$ Sample the challenge constrained key during setup instead of during the challenge phase. Then, during key generation we reverse engineer the secret keys to retain the additive distribution we are looking for.

Setup $(1^K, 1^\kappa, f, U)$ The challenger samples $\mathsf{crs} \leftarrow \mathsf{ppGen}(1^\lambda, 1^{|f|})$, $\mathsf{sk} \leftarrow \mathsf{skGen}(\mathsf{crs})$ and $\mathsf{ck}_f \leftarrow \mathsf{Constrain}(\mathsf{sk}, f)$. Then the challenger samples $r^{(0)}, r^{(1)} \leftarrow \{0,1\}^\lambda$ and finds a $k \in [\lambda]$ such that $r_k^{(0)} \neq r_k^{(1)}$. The challenger chooses any honest authority index $h \in [K]\setminus U$. Finally the challenger generates $2K\lambda$ many secret keys $\forall\ i \in [K],\ j \in [\lambda],\ b \in \{0,1\} : \mathsf{sk}_{i,j,b} \leftarrow \mathsf{skGen}(\mathsf{crs})$, and sends $(\mathsf{crs}, \{\mathsf{sk}_{i,j,b}\}_{i\in U, j\in[\lambda], b\in\{0,1\}})$ to $\mathcal{A}$.

Query $\mathsf{SkEnc}(f^*)$ The challenger calculates $\mathsf{sk}_1 = \sum_{i\in[K]} \sum_{j\in[\lambda]} \mathsf{sk}_{i,j,r_j^{(1)}}$ and $\mathsf{ck}_1 \leftarrow \mathsf{Constrain}(\mathsf{sk}_1, f^*)$. Then responds with $\mathsf{ct}_1 = (r^{(1)}, \mathsf{ck}_1)$.

Query $\mathsf{KGen}(i,x)$ Respond with $\{\mathsf{Eval}(\mathsf{sk}_{i,j,b}, x)\}_{j\in[\lambda],b\in\{0,1\}}$, if authority h is queried then we replace $\mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ in our response, instead, we put $E_{h,k,x} = \mathsf{Eval}(\mathsf{sk}, x) - \mathsf{Eval}(\sum_{i\in[K],j\in[\lambda]} \mathsf{sk}_{i,j,r_j^{(0)}}) + \mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$

Challenge Ciphertext Send $\mathsf{ct} = (r^{(0)}, \mathsf{ck}_f)$.

$\mathsf{Hybrid}^{\mathcal{A}}(2)$ On key generation queries where $f^{(b)}(x) = 1$ we respond normally, on queries where $f^{(b)}(x) = 0$ we use the constrained key to instead to evaluate the secret key query.

Setup $(1^K, 1^\kappa, f, U)$ The challenger samples $\mathsf{crs} \leftarrow \mathsf{ppGen}(1^\lambda, 1^{|f|})$, $\mathsf{sk} \leftarrow \mathsf{skGen}(\mathsf{crs})$ and $\mathsf{ck}_f \leftarrow \mathsf{Constrain}(\mathsf{sk}, f)$. Then the challenger samples $r^{(0)}, r^{(1)} \leftarrow \{0,1\}^\lambda$ and finds a $k \in [\lambda]$ such that $r_k^{(0)} \neq r_k^{(1)}$. The challenger chooses any honest authority index $h \in [K] \backslash U$. Finally the challenger generates $2K\lambda$ many secret keys $\forall\ i \in [K],\ j \in [\lambda],\ b \in \{0,1\} : \mathsf{sk}_{i,j,b} \leftarrow \mathsf{skGen}(\mathsf{crs})$, and sends $(\mathsf{crs}, \{\mathsf{sk}_{i,j,b}\}_{i \in U, j \in [\lambda], b \in \{0,1\}})$ to $\mathcal{A}$.

Query $\mathsf{SkEnc}(f^*)$ The challenger calculates $\mathsf{sk}_1 = \sum_{i \in [K]} \sum_{j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(1)}}$ and $\mathsf{ck}_1 \leftarrow \mathsf{Constrain}(\mathsf{sk}_1, f^*)$. Then responds with $\mathsf{ct}_1 = (r^{(1)}, \mathsf{ck}_1)$.

Query $\mathsf{KGen}(i, x)$ Respond with $\{\mathsf{Eval}(\mathsf{sk}_{i,j,b}, x)\}_{j \in [\lambda], b \in \{0,1\}}$, if authority h is queried then we replace $\mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ in our response, instead, we put $E_{h,k,x} = \mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x) - \mathsf{Eval}(\sum_{i \in [K], j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(0)}}) + \mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ when $f(x) = 0$ and $E_{h,k,x} = \mathsf{Eval}(\mathsf{sk}, x) - \mathsf{Eval}(\sum_{i \in [K], j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(0)}}) + \mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ when $f(x) = 1$.

Challenge Ciphertext Send $\mathsf{ct} = (r^{(0)}, \mathsf{ck}_f)$.

$\mathsf{Hybrid}^{\mathcal{A}}(3)$ We replace the constrained key generated during setup with a simulated constrained key. This, in turn, makes key generation queries pseudorandom.

Setup $(1^K, 1^\kappa, f, U)$ The challenger samples $\mathsf{ck}_f \leftarrow \mathsf{SIM}.(1^\lambda, 1^{|f|})$. Then the challenger samples $r^{(0)}, r^{(1)} \leftarrow \{0,1\}^\lambda$ and finds a $k \in [\lambda]$ such that $r_k^{(0)} \neq r_k^{(1)}$. The challenger chooses any honest authority index $h \in [K] \backslash U$. Finally the challenger generates $2K\lambda$ many secret keys $\forall\ i \in [K],\ j \in [\lambda],\ b \in \{0,1\} : \mathsf{sk}_{i,j,b} \leftarrow \mathsf{skGen}(\mathsf{crs})$, and sends $(\mathsf{crs}, \{\mathsf{sk}_{i,j,b}\}_{i \in U, j \in [\lambda], b \in \{0,1\}})$ to $\mathcal{A}$.

Query $\mathsf{SkEnc}(f^*)$ The challenger calculates $\mathsf{sk}_1 = \sum_{i \in [K]} \sum_{j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(1)}}$ and $\mathsf{ck}_1 \leftarrow \mathsf{Constrain}(\mathsf{sk}_1, f^*)$. Then responds with $\mathsf{ct}_1 = (r^{(1)}, \mathsf{ck}_1)$.

Query $\mathsf{KGen}(i, x)$ Respond with $\{\mathsf{Eval}(\mathsf{sk}_{i,j,b}, x)\}_{j \in [\lambda], b \in \{0,1\}}$, if authority h is queried then we replace $\mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ in our response, instead, we put $E_{h,k,x} = \mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x) - \mathsf{Eval}(\sum_{i \in [K], j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(0)}}) + \mathsf{Eval}(\mathsf{sk}_{h,k,r_k^{(0)}}, x)$ when $f(x) = 0$ and $E_{h,k,x} \leftarrow U(\mathcal{R}_\lambda)$ when $f(x) = 1$.

Challenge Ciphertext Send $\mathsf{ct} = (r^{(0)}, \mathsf{ck}_f)$.

$\mathsf{Hybrid}^{\mathcal{A}}(4)$ The real security game in the public-key Enc mode.

Setup $(1^K, 1^\kappa, f, U)$ The challenger samples $\mathsf{ck}_f \leftarrow \mathsf{SIM}.(1^\lambda, 1^{|f|})$. Then the challenger samples $r^{(0)}, r^{(1)} \leftarrow \{0,1\}^\lambda$, stopping if $r^{(0)} = r^{(1)}$. Finally the challenger generates $2K\lambda$ many secret keys $\forall\ i \in [K],\ j \in [\lambda],\ b \in \{0,1\} : \mathsf{sk}_{i,j,b} \leftarrow \mathsf{skGen}(\mathsf{crs})$, and sends $(\mathsf{crs}, \{\mathsf{sk}_{i,j,b}\}_{i \in U, j \in [\lambda], b \in \{0,1\}})$ to $\mathcal{A}$.

Query $\mathsf{SkEnc}(f^*)$ The challenger calculates $\mathsf{sk}_1 = \sum_{i \in [K]} \sum_{j \in [\lambda]} \mathsf{sk}_{i,j,r_j^{(1)}}$ and $\mathsf{ck}_1 \leftarrow \mathsf{Constrain}(\mathsf{sk}_1, f^*)$. Then responds with $\mathsf{ct}_1 = (r^{(1)}, \mathsf{ck}_1)$.

Query $\mathsf{KGen}(i, x)$ Respond with $\{\mathsf{Eval}(\mathsf{sk}_{i,j,b}, x)\}_{j \in [\lambda], b \in \{0,1\}}$.

Challenge Ciphertext Send $\mathsf{ct} = (r^{(0)}, \mathsf{ck}_f)$.

Lemma 6.10. Assuming the Functionality Preservation, Almost Key-Homorphism properties of our PCPRF scheme, $\mathsf{Hybrid}^{\mathcal{A}}(0)$ and $\mathsf{Hybrid}^{\mathcal{A}}(1)$ are statistically indistinguishable

Proof. These two hybrids are identical to $\mathsf{Hybrid}^{\mathcal{A}}(0,b)$ and $\mathsf{Hybrid}^{\mathcal{A}}(1,b)$ from the Function Indistinguishability proof, except $f^{(b)}$ is replaced by f. See Function Indistinguishability $\mathsf{Hybrid}^{\mathcal{A}}(0,b) \to \mathsf{Hybrid}^{\mathcal{A}}(1,b)$ on page 29. □

Lemma 6.11. Assuming the Functionality Preservation property of our PCPRF scheme, $\mathsf{Hybrid}^{\mathcal{A}}(1)$ and $\mathsf{Hybrid}^{\mathcal{A}}(2)$ are statistically indistinguishable.

Proof. This is simply the application of the Functionality Preservation property. □

Lemma 6.12. Assuming the Pseudorandomness and Constraint-Hiding property of our PCPRF scheme, $\mathsf{Hybrid}^{\mathcal{A}}(2)$ and $\mathsf{Hybrid}^{\mathcal{A}}(3)$ are computationally indistinguishable.

Proof. If an attacker $\mathcal{A}$ is able to distinguish between $\mathsf{Hybrid}^{\mathcal{A}}(2)$ and $\mathsf{Hybrid}^{\mathcal{A}}(3)$ then they would have to distinguish between the real and simulated constrained keys in the Pseudorandomness and Constraint-Hiding game. □

Lemma 6.13. Assuming $\mathcal{A}$ is a valid Accept Indistinguishability attacker then $\mathsf{Hybrid}^{\mathcal{A}}(3)$ and $\mathsf{Hybrid}^{\mathcal{A}}(4)$ are computationally indistinguishable.

Proof. If we only consider x where $f(x) = 1$, then $E_{h,k,x}$ will be randomly sampled in $\mathsf{Hybrid}^{\mathcal{A}}(3)$ and $\mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x)$ will look pseudorandom with respect to all secret key evaluations in $\mathsf{Hybrid}^{\mathcal{A}}(4)$. In both cases the distribution of $\mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x) - \sum_{i\in[K],j\in[\lambda]} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j^{(0)}}, x)$ will look random. If we only consider x where $f(x) = 0$ then we know any valid Accept Indistinguishability adversary cannot query all $\{(i,x)\}_{i\in[K]\setminus U}$ in the key generation phase. This means that $\mathcal{A}$ will not have enough information to calculate $\sum_{i\in[K],j\in[\lambda]} \mathsf{Eval}(\mathsf{sk}_{i,j,r_j^{(0)}}, x)$ and thus will have no information about how close this is to $\mathsf{CEval}(\mathsf{ck}_{f^{(b)}}, x)$. Meaning, for these x, $\mathcal{A}$ will not be able to tell if the secret key evaluations were generated separately from the constrained key as in the real game ($\mathsf{Hybrid}^{\mathcal{A}}(4)$) or with respect to the constrained key as in $\mathsf{Hybrid}^{\mathcal{A}}(3)$. □

To conclude the Accept Indistinguishability proof, we can follow the hybrid indistinguishability from $\mathsf{Hybrid}^{\mathcal{A}}(0) \to \mathsf{Hybrid}^{\mathcal{A}}(1) \to \mathsf{Hybrid}^{\mathcal{A}}(2) \to \mathsf{Hybrid}^{\mathcal{A}}(3) \to \mathsf{Hybrid}^{\mathcal{A}}(4)$. So the real game in the secret-key encryption mode ($\mathsf{Hybrid}^{\mathcal{A}}(0)$) must be indistinguishable from the real game in the public-key encryption mode ($\mathsf{Hybrid}^{\mathcal{A}}(4)$).

7 Alternate Constructions for MA-TT

7.1 Preliminaries

In this section, we provide the definitions of Multi-Authority Functional Encryption (MAFE) and Identity-Based Encryption (IBE) which we use in our alternate constructions.

Multi-Authority Functional Encryption. A Multi-Authority Functional Encryption (MAFE) scheme for global identifier space $\mathsf{GID} = \{\mathsf{GID}_N\}_N$ and circuit class $\mathcal{C} = \{\mathcal{C}_\kappa : (\mathcal{X}_\kappa)^K \to \mathcal{Y}_\kappa\}_{\kappa,K}$ consists of the following polynomial time algorithms.

$\mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa, 1^N)$ The randomized global setup algorithm takes as input security parameter λ, number of authorities K, functionality index κ, number of global identities N, and outputs common reference string crs, which is assumed to be an input of all the following functions.

$\mathsf{ASetup}(i) \to (\mathsf{mpk}_i, \mathsf{msk}_i)$ The randomized authority setup algorithm takes as input an authority index $i \in [K]$, and outputs associated master secret key msk_i and master public key mpk_i.

$\mathsf{KGen}(i, \mathsf{msk}_i, \mathsf{gid}, x) \to \mathsf{sk}_{i,\mathsf{gid},x}$ The randomized key generation algorithms takes as input an authority index i along with an associated master secret key msk_i. Then it uses the inputted user identity $\mathsf{gid} \in \mathsf{GID}_N$ and circuit input $x \in \mathcal{X}_\kappa$ to generate a user secret key associated with authority i, user gid, and circuit input x called $\mathsf{sk}_{i,\mathsf{gid},x}$.

$\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, f) \to \mathsf{ct}_f$ The randomized public key encryption algorithm takes as input all authority master public keys msk_i and circuit f, and returns a ciphertext ct_f.

$\mathsf{Dec}(\{\mathsf{sk}_{i,\mathsf{gid},x_i}\}_{i\in[K]}, \mathsf{ct}_f) \to y$ The deterministic decryption algorithm takes a ciphertext ct_f and a set of secret keys $\mathsf{sk}_{i,\mathsf{gid},x_i}$ generated by each authority index under the same user id gid and some circuit input x_i. Then it returns circuit output $y \in \mathcal{Y}_\kappa$.

Definition 7.1. A Multi-Authority Functional Encryption scheme (GSetup, ASetup, KGen, Enc, Dec) must satisfy the following properties.

Correctness There exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $K = \mathsf{poly}(\lambda)$, $\kappa \in \mathsf{poly}(\lambda)$, $N \in \mathsf{poly}(\lambda)$, $\{x_i\}_{i\in[K]} \in (\mathcal{X}_\kappa)^K$, $f \in \mathcal{C}_\kappa$,

$$\Pr\left[\begin{array}{c|c|l} & & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa, 1^N) \\ & & \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ f(\{x_i\}_{i\in[K]}) = y & : & \forall\ i \in [K], \mathsf{sk}_{i,\mathsf{gid},x_i} \leftarrow \mathsf{KGen}(i, \mathsf{msk}_i, \mathsf{gid}, x_i) \\ & & \mathsf{ct}_f \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, f) \\ & & y = \mathsf{Dec}(\{\mathsf{sk}_{i,\mathsf{gid},x_i}\}_{i\in[K]}, \mathsf{ct}_f) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Adaptive Security For any admissible PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$:

$$\Pr\left[\begin{array}{c|c|l} & & (1^K, 1^\kappa, 1^N) \leftarrow \mathcal{A}(1^\lambda) \\ & & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^\kappa, 1^N) \\ & & \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ & & (\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}) \to \mathcal{A} \\ b' = b & : & (f^{(0)}, f^{(1)}) \leftarrow \mathcal{A}^{\mathsf{KGen}(\cdot,\mathsf{msk}_{(\cdot)},\cdot,\cdot),\mathsf{Corr}(\cdot)} \\ & & b \leftarrow \{0,1\} \\ & & \mathsf{ct}_b \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, f^{(b)}) \\ & & b' \leftarrow \mathcal{A}^{\mathsf{KGen}(\cdot,\mathsf{msk}_{(\cdot)},\cdot,\cdot),\mathsf{Corr}(\cdot)}(\mathsf{ct}_b) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

For each complete set $\{(i, \mathsf{gid}, x_i)\}_{i\in[K]}$ queried by $\mathcal{A}$ in the KGen oracle, it must be true that $f^{(0)}(x_1, ..., x_a) = f^{(1)}(x_1, ..., x_a)$.

The corruption oracle Corr takes as input an authority index i and returns the corresponding master secret key msk_i.

Identity-Based Encryption. An Identity Based Encryption scheme with message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_\lambda$ and user identity space $\mathsf{GID} = \{\mathsf{GID}_N\}_N$ consists of the following polynomial time algorithms.

$\mathsf{Setup}(1^\lambda, 1^N) \to (\mathsf{mpk}, \mathsf{msk})$ The randomized setup algorithm takes as input a security parameter λ and returns a master public key mpk and master secret key msk. The master public key mpk is assumed as input to the following algorithms.

$\mathsf{KGen}(\mathsf{msk}, \mathsf{gid}) \to \mathsf{sk}_{\mathsf{gid}}$ The randomized key generation algorithm takes as input the master secret key msk and user identity $\mathsf{gid} \in \mathsf{GID}_N$ and returns the corresponding secret key $\mathsf{sk}_{\mathsf{gid}}$.

$\mathsf{Enc}(\mathsf{mpk}, \mathsf{gid}, m) \to \mathsf{ct}$ The randomized encryption algorithm takes as input a user identity $\mathsf{gid} \in \mathsf{GID}_N$ and message $m \in \mathcal{M}_\lambda$ then outputs a ciphertext ct.

$\mathsf{Dec}(\mathsf{ct}, \mathsf{sk}_{\mathsf{gid}}) \to m/\perp$ The deterministic decryption algorithm takes as input ciphertext ct and secret key $\mathsf{sk}_{\mathsf{gid}}$, and returns message m, or failure character $\perp$ when decryption fails.

Definition 7.2. An Identity-Based Encryption scheme $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ must satisfy the following properties.

Correctness There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$, $m \in \mathcal{M}_\lambda$,

$$\Pr\left[\begin{array}{c|cl} \mathsf{Dec}(\mathsf{ct}, \mathsf{sk}_{\mathsf{gid}}) = m & : & (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ & & \mathsf{sk}_{\mathsf{gid}} \leftarrow \mathsf{KGen}(\mathsf{msk}, \mathsf{gid}) \\ & & \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathsf{gid}, m) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Selective Identity IND-CPA Security For any admissible PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$:

$$\Pr\left[\begin{array}{c|cl} b' = b & : & (1^N, \mathsf{gid}) \leftarrow \mathcal{A}(1^\lambda) \\ & & (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^N) \\ & & (m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{KGen}(\mathsf{msk}, \cdot)}(\mathsf{mpk}, \mathsf{ct}) \\ & & b \leftarrow \{0, 1\} \\ & & \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathsf{gid}, m_b) \\ & & b' \leftarrow \mathcal{A}^{\mathsf{KGen}(\mathsf{msk}, \cdot)}(\mathsf{ct}) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

$\mathcal{A}$ cannot query gid in the KGen oracle.

7.2 MA-PLBE with Public Key Index Encryption Definition

Syntax. We define a slightly altered notion of Multi-Authority Private Linear Broadcast Encryption by making the $\mathsf{EncIndex}$ algorithm a public-key algorithm (and thus remove the traditional Enc algorithm).

$\mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i \in [K]}, j, m) \to \mathsf{ct}$ The randomized public key index encryption algorithm takes as input all authority master public keys msk_i the message m, and a user index $j \in [0, ..., N]$. Returning a ciphertext ct.

Definition 7.3. A MA-PLBE scheme with Public Key Index Encryption $(\mathsf{GSetup}, \mathsf{ASetup}, \mathsf{KGen}, \mathsf{EncIndex}, \mathsf{Dec})$ must satisfy the same security properties as a traditional MA-PLBE scheme, however the Normal Hiding Security property becomes unnecessary. See MA-PLBE definition on page 15 for details.

Theorem 7.4. Assuming the existence of a MA-PLBE scheme with Public Key Index Encryption: $\mathsf{PLBE} = (\mathsf{GSetup}, \mathsf{ASetup}, \mathsf{KGen}, \mathsf{EncIndex}, \mathsf{Dec})$ then there exists an MA-TT scheme: $\mathsf{MATT} = (\mathsf{GSetup}, \mathsf{ASetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$ with a public-key tracing algorithm.

Proof. Outline: Decoder PLBE with Public Key Index Encryption follows immediately from this definition since the reduction algorithm can simply sample their own ciphertexts when testing the decoder box instead of relying on 1-Bounded queries. Then notice that in the construction of MA-TT from Decoder-PLBE on page 50, the tracing algorithm only uses master secret keys to generate ciphertexts. So in the Public Key Index Encryption mode, tracing would only require master public keys. So the theorem follows from using the exact construction of MA-TT from MA-PLBE used in this paper. □

Remark 7.5. We will provide two constructions for MA-PLBE with Public Key Index Encryption in the following sections, one from Multi-Authority Functional Encryption, and the other from Identity-Based Encryption.

7.3 MA-PLBE with Public Key Index Encryption from MA-FE

MA-FE is a significantly more powerful object compared to MA-PLBE with Public Key Index Encryption. This construction simply uses a comparison function as the MA-FE encrypted function.

$\mathsf{GSetup}(1^\lambda, 1^K, 1^N) \to \mathsf{crs}$ Set the MA-FE function class to the set of functions that go from a set of user indexes to the message space.

Let $C_\kappa : ([N])^K \to (\mathcal{M}_\lambda \cup \{\perp\})$

Let $\mathsf{crs} \leftarrow \mathsf{FE.GSetup}(1^\lambda, 1^K, 1^\kappa, 1^N)$

$\mathsf{ASetup}(i) \to (\mathsf{mpk}_i, \mathsf{msk}_i)$ Perform the MA-FE authority setup algorithm.

Let $(\mathsf{mpk}_i, \mathsf{msk}_i) = (\mathsf{FE.mpk}_i, \mathsf{FE.msk}_i) \leftarrow \mathsf{FE.ASetup}(i)$.

$\mathsf{KGen}(i, \mathsf{msk}_i, j) \to \mathsf{sk}_{i,j}$ Create a MA-FE secret key with the MA-FE key generation algorithm.

Let $\mathsf{sk}_{i,j} = \mathsf{FE.sk}_{i,j,j} \leftarrow \mathsf{FE.KGen}(i, \mathsf{msk}_i, j, j)$.

$\mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, j, m) \to \mathsf{ct}$ Encrypt a function that compares the input to j, then either outputs m or $\perp$. Sample $ct \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, \mathsf{comp}_{j,m}(\cdot))$ where $\mathsf{comp}_{j,m}(\cdot)$ is defined below.

$\mathsf{comp}_{j,m}(\{x_i\}_{i\in[K]})$:

If $(x_1 = x_2 = ... = x_a) \wedge (x_1 \overset{?}{>} j) \to m$

Else $\to \perp$

$\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct}) \to m/\perp$

Theorem 7.6. If FE is a valid Multi-Authority Functional Encryption scheme then the algorithms (GSetup, ASetup, KGen, EncIndex, Dec) constitute a valid MA-PLBE scheme with Public Key Index Encryption.

Proof.

Correctness For any $\lambda \in \mathbb{N}$, $K = \mathsf{poly}(\lambda)$, $N \in \mathsf{poly}(\lambda)$, $m \in \mathcal{M}_\lambda, j, k \in [N]$

$$\mathsf{FE.crs} \leftarrow \mathsf{FE.GSetup}(1^\lambda, 1^K, 1^{\mathsf{FE}.\kappa}, 1^N),$$
$$\forall\ i \in [K] : (\mathsf{FE.mpk}_i, \mathsf{FE.msk}_i) \leftarrow \mathsf{ASetup}(i)$$
$$\forall\ i \in [K] : \mathsf{FE.sk}_{i,k,k} \leftarrow \mathsf{FE.KGen}(i, \mathsf{FE.msk}_i, k, k)$$
$$\mathsf{ct} \leftarrow \mathsf{FE.Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, \mathsf{comp}_{k,m})$$

Correctness follows directly from MA-FE correctness.

$$\mathsf{FE.Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{FE.Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, \mathsf{comp}_{k,m})) = \mathsf{comp}_{k,m}(\{j\}_{i\in[K]})$$

Where the value $\mathsf{comp}_{k,m}(\{j\}_{i\in[K]}) = m$ when $j > k$.

Index Hiding Security Consider adversary $\mathcal{A}$ to the Index Hiding Security of our MA-PLBE scheme with Public Key Index Encryption, and challenger $\mathcal{C}$ for the MA-FE Adaptive Security game. Reduction $\mathcal{R}$ is as follows:

Setup $(1^K, N, j)$ $\mathcal{R}$ initializes the challenger $\mathcal{C}$ by sending $(1^K, 1^{|\mathsf{comp}|}, 1^N)$. Then $\mathcal{R}$ will receive $(\mathsf{FE.crs}, \{\mathsf{FE.mpk}_i\}_{i\in[K]}) \leftarrow \mathcal{C}$, and $\mathcal{R}$ forwards $(\mathsf{FE.crs}, \{\mathsf{FE.mpk}_i|\}_{i\in[K]}) \rightarrow \mathcal{A}$.

Key Generation (i, j) $\mathcal{R}$ sends (i, j, j) to $\mathcal{C}$ as a KGen oracle query. $\mathcal{C}$ responds with $\mathsf{sk}_{i,j,j}$ which $\mathcal{R}$ forwards to $\mathcal{A}$.

Corruption Oracle (i) $\mathcal{R}$ sends (i) to the challenger's corruption oracle and receives $\mathsf{FE.msk}_i$ from $\mathcal{C}$. Then $\mathcal{R}$ forwards $\mathsf{FE.msk}_i \rightarrow \mathcal{C}$.

Challenge Phase (m) $\mathcal{R}$ sends $(\mathsf{comp}_{j,m}, \mathsf{comp}_{j-1,m})$ to $\mathcal{C}$ as the challenge functions, receiving ct_b. Then $\mathcal{R}$ sends ct_b to $\mathcal{A}$, and $\mathcal{A}$ responds with b' which $\mathcal{R}$ forwards directly to the challenger $\mathcal{C}$.

The only set of $\{x_i\}_{i\in[K]}$ where $\mathsf{comp}_{j,m}(\{x_i\}_{i\in[K]})$ differs from $\mathsf{comp}_{j-1,m}(\{x_i\}_{i\in[K]})$ is exactly the set $\{j\}_{i\in[K]}$. However, $\mathcal{A}$ cannot query all $\{(i, j)\}_{i\in[K]}$ in the key generation phase, so $\mathcal{R}$ must also be an admissible adversary in the MA-FE Adaptive Security game.
If $\mathcal{A}$ has advantage ϵ in the Index Hiding Security game, then $\mathcal{R}$ must have advantage ϵ in the MA-FE Adaptive Security game. This is a contradiction of the validity of our MA-FE scheme, so the construction on page 35 must have the Index Hiding Security property.

Message Hiding Security We create three hybrids, $\mathsf{Hybrid}^{\mathcal{A}}(0)$ is the normal security game when the challenger responds with $\mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, N, m_0)$, $\mathsf{Hybrid}^{\mathcal{A}}(1)$ is the normal security game when the challenger responds with $\mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, N, m_1)$, and $\mathsf{Hybrid}^{\mathcal{A}}(2)$ is a mode where the challenger responds with $\mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, N, \perp)$. For Message Hiding Security we want to prove that $\mathsf{Hybrid}^{\mathcal{A}}(0)$ is indistinguishable from $\mathsf{Hybrid}^{\mathcal{A}}(1)$, which we will do by proving that for $b \in \{0, 1\}$, $\mathsf{Hybrid}^{\mathcal{A}}(b)$ is indistinguishable from $\mathsf{Hybrid}^{\mathcal{A}}(2)$.
For $b \in \{0, 1\}$, suppose there exists an adversary $\mathcal{A}$ that can distinguish $\mathsf{Hybrid}^{\mathcal{A}}(b)$ from $\mathsf{Hybrid}^{\mathcal{A}}($ 2). Let challenger $\mathcal{C}$ be for the MA-FE Adaptive Security game. Then reduction $\mathcal{R}$ is as follows:

Setup $(1^K, N)$ $\mathcal{R}$ initializes the challenger $\mathcal{C}$ by sending $(1^K, 1^{|\mathsf{comp}|}, 1^N)$. Then $\mathcal{R}$ will receive $(\mathsf{FE.crs}, \{\mathsf{FE.mpk}_i\}_{i\in[K]}) \leftarrow \mathcal{C}$, and $\mathcal{R}$ forwards $(\mathsf{FE.crs}, \{\mathsf{FE.mpk}_i|\}_{i\in[K]}) \rightarrow \mathcal{A}$.

Key Generation (i, j) $\mathcal{R}$ sends (i, j, j) to $\mathcal{C}$ as a KGen oracle query. $\mathcal{C}$ responds with $\mathsf{sk}_{i,j,j}$ which $\mathcal{R}$ forwards to $\mathcal{A}$.

Corruption Oracle (i) $\mathcal{R}$ sends (i) to the challenger's corruption oracle and receives $\mathsf{FE.msk}_i$ from $\mathcal{C}$. Then $\mathcal{R}$ forwards $\mathsf{FE.msk}_i \to \mathcal{C}$.

Challenge Phase (m_0, m_1) $\mathcal{R}$ sends $(\mathsf{comp}_{N,m_b}, \mathsf{comp}_{N,\perp})$ to $\mathcal{C}$ as the challenge functions, receiving ct_b. Then $\mathcal{R}$ sends ct_b to $\mathcal{A}$, and $\mathcal{A}$ responds with b' which $\mathcal{R}$ forwards directly to the challenger $\mathcal{C}$.

The only $\{x_i\}_{i\in[K]}$ where $\mathsf{comp}_{N,m_b}(\{x_i\}_{i\in[K]})$ differs from $\mathsf{comp}_{N,\perp}(\{x_i\}_{i\in[K]})$ would be a set $\{M\}_{i\in[K]}$ where $M > N$. However, (i, M) would be an invalid secret key query in the Message Hiding Security game, so $\mathcal{R}$ will be an admissible adversary in the MA-FE Adaptive Security game. If $\mathcal{A}$ has advantage ϵ when distinguishing $\mathsf{Hybrid}^{\mathcal{A}}(b)$ from $\mathsf{Hybrid}^{\mathcal{A}}(2)$, then $\mathcal{R}$ must have advantage ϵ in the MA-FE Adaptive Security game. This is a contradiction, so there cannot exist an adversary who can distinguish $\mathsf{Hybrid}^{\mathcal{A}}(b)$ from $\mathsf{Hybrid}^{\mathcal{A}}(2)$. So there also cannot exist an adversary that can distinguish $\mathsf{Hybrid}^{\mathcal{A}}(0)$ from $\mathsf{Hybrid}^{\mathcal{A}}(1)$. This is exactly the Message Hiding Security game, so the construction on page 35 must satisfy the Message Hiding Security property.

□

7.4 Multi-Authority Index Encryption from IBE

The idea behind this construction is to have each authority create their own Identity-Based Encryption scheme, then when we want to encrypt a message we create $K * N$ many IBE encryptions, one for each authority and user combination. Inside of each set of K many encryptions under the same identity we put a share of the message m or $\overline{0} = \perp$ depending on the index we are trying to encrypt.

$\mathsf{GSetup}(1^\lambda, 1^K, 1^N) \to \mathsf{crs}$

Let $\mathsf{crs} = (1^\lambda, 1^K, 1^N)$

$\mathsf{ASetup}(i) \to (\mathsf{mpk}_i, \mathsf{msk}_i)$ Setup an IBE scheme for this authority and return that scheme's master public key and master secret key.

Let $(\mathsf{mpk}_i, \mathsf{msk}_i) = (\mathsf{IBE.mpk}, \mathsf{IBE.msk}) \leftarrow \mathsf{IBE.Setup}(1^\lambda, 1^N)$.

$\mathsf{KGen}(i, \mathsf{msk}_i, j) \to \mathsf{sk}_{i,j}$ Create an IBE secret key with $\mathsf{gid} = j$.

Let $\mathsf{sk}_{i,j} = \mathsf{IBE.sk}_{i,j} \leftarrow \mathsf{IBE.KGen}(\mathsf{msk}_i, j)$.

$\mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, j, m) \to \mathsf{ct}$ First generate randomness for each authority and user index $\forall i \in [K-1], j \in [N] : r_{i,j} \leftarrow \{0,1\}^{|m|}$. Then make it so that each set of user indexes k sum to either the message m if $k > j$ or sum to $\overline{0}$ when $k \leq j$.

Calculate $\forall\, k \in [N], k \leq j : r_{K,k} = \overline{0} \bigoplus_{i\in[K-1]} r_{i,k}$

Calculate $\forall\, k \in [N], k > j : r_{K,k} = m \bigoplus_{i\in[K-1]} r_{i,k}$

Calculate $\mathsf{ct}_{i,j} \leftarrow \mathsf{IBE.Enc}(\mathsf{mpk}_i, j, r_{i,j})$, then let $\mathsf{ct} = \{\mathsf{ct}_{i,j}\}_{i\in[K],j\in[N]}$

$\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct}) \to m/\perp$ Interpret $\mathsf{ct} = \{\mathsf{ct}_{i,j}\}_{i\in[K],j\in[N]}$ and only consider $\{\mathsf{ct}_{i,j}\}_{i\in[K]}$. We use IBE to decrypt each ciphertext using it's corresponding secret key and then combine random shares to receive the message or $\perp$.

Calculate $\forall\, i \in [K] : r_{i,j} = \mathsf{IBE.Dec}(\mathsf{ct}_{i,j}, \mathsf{sk}_{i,j})$

Output $\bigoplus_{i\in[K]} r_{i,j}$

Theorem 7.7. If IBE is a valid Identity-Based Encryption scheme then the algorithms (GSetup, ASetup, KGen, EncIndex, Dec) constitute a valid Multi-Authority Index Encryption scheme.

Correctness For any $\lambda \in \mathbb{N}$, $K = \mathsf{poly}(\lambda)$, $N \in \mathsf{poly}(\lambda)$, $m \in \mathcal{M}_\lambda, j, k \in [N]$

$$(1^\lambda, 1^K, 1^N) \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N), \forall\ i \in [K]$$
$$(\mathsf{IBE.mpk}_i, \mathsf{IBE.msk}_i) \leftarrow \mathsf{ASetup}(i)$$
$$\forall\ i \in [K], \mathsf{IBE.sk}_{i,j} \leftarrow \mathsf{KGen}(i, \mathsf{msk}_i, j)$$
$$\{\mathsf{IBE.Enc}(\mathsf{mpk}_i, j, r_{i,j,(\cdot)})\}_{i\in[K], j\in[N]} \leftarrow \mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, k, m)$$

The first step of decryption is to use IBE.Dec, according to IBE correctness we have with probability greater than $1 - \mathsf{negl}(\lambda)$ that $\mathsf{IBE.Dec}(\mathsf{IBE.Enc}(\mathsf{mpk}_i, j, r_{i,k}, \mathsf{IBE.sk}_{i,j}) = r_{i,j}$. So with probability greater than $1 - K * \mathsf{negl}(\lambda)$, decryption will successfully calculate $\{r_{i,j,(\cdot)}\}_{i\in[K]}$. Then we can simply combine these $r_{i,j}$ by $m = \bigoplus_{i\in[K]} r_{i,j}$

Index Hiding Security For this proof we will need to create four hybrids:

$\mathsf{Hybrid}^{\mathcal{A}}(0)$ The challenge ciphertext is generated normally by $\mathsf{ct} \leftarrow \mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, j, m)$

$\mathsf{Hybrid}^{\mathcal{A}}(1)$ The challenge ciphertext is generated normally by $\mathsf{ct} \leftarrow \mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, j, m)$ except for whatever authority index i^* that isn't queried by the corruption oracle or the key generation oracle on user j, $\mathsf{ct}_{i^*,j} = \mathsf{IBE.Enc}(\mathsf{IBE.mpk}_{i^*}, j, 0)$.

$\mathsf{Hybrid}^{\mathcal{A}}(2)$ The challenge ciphertext is generated by $\mathsf{ct} \leftarrow \mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, j-1, m)$ except for whatever authority index i^* that isn't queried by the corruption oracle or the key generation oracle on user j, $\mathsf{ct}_{i^*,j} = \mathsf{IBE.Enc}(\mathsf{IBE.mpk}_{i^*}, j, 0)$.

$\mathsf{Hybrid}^{\mathcal{A}}(3)$ The challenge ciphertext is generated by $\mathsf{ct} \leftarrow \mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, j-1, m)$.

For $\mathsf{Hybrid}^{\mathcal{A}}(0) \to \mathsf{Hybrid}^{\mathcal{A}}(1)$ and $\mathsf{Hybrid}^{\mathcal{A}}(2) \to \mathsf{Hybrid}^{\mathcal{A}}(3)$ if an attacker can distinguish between the two ciphertexts then they would need to distinguish between $\mathsf{Enc}(\mathsf{IBE.mpk}_{i^*}, j, r_{i,j})$ and $\mathsf{Enc}(\mathsf{IBE.mpk}_{i^*}, j, 0)$. However, we know that i^* is an uncorrupted authority, and the adversary never queried for $\mathsf{gid} = j$ under this authority. This would break IBE's Selective Identity IND-CPA Security.

For $\mathsf{Hybrid}^{\mathcal{A}}(1) \to \mathsf{Hybrid}^{\mathcal{A}}(2)$ if an attacker can distinguish here then they would need to distinguish between $\{r_{i,j}\}_{i\in[K]\setminus i^*}$ when $\bigoplus_{i\in[K]} r_{i,j} = m$ and $\{r'_{i,j}\}_{i\in[K]\setminus i^*}$ when $\bigoplus_{i\in[K]} r_{i,j} = \overline{0}$. However, these distributions are statistically indistinguishable since they are both missing the last piece of information necessary to reconstruct m and $\overline{0}$ respectively.

Message Hiding Security Notice that $\mathsf{ct} \leftarrow \mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, N, m)$ contains no information about m itself. So it is immediately clear that $\mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, N, m_0) \approx_s \mathsf{EncIndex}(\{\mathsf{mpk}_i\}_{i\in[K]}, N, m_1)$ even for an attacker that corrupts every authority.

References

[AG21] Miguel Ambrona and Romain Gay. Multi-authority abe, revisited. *Cryptology ePrint Archive*, 2021.

[AGT21] Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-party functional encryption. In *Theory of Cryptography: 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8–11, 2021, Proceedings, Part II*, pages 224–255. Springer, 2021.

[AKYY23] Shweta Agrawal, Simran Kumari, Anshu Yadav, and Shota Yamada. Broadcast, trace and revoke with optimal parameters from polynomial hardness. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part III*, volume 14006 of *Lecture Notes in Computer Science*, pages 605–636, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

[BCC+25] Elaine Barker, Lily Chen, David Cooper, Dustin Moody, Andrew Regenscheid, Murugiah Souppaya, Bill Newhouse, Russ Housley, Sean Turner, William Barker, et al. Considerations for achieving crypto agility. *Cybersecurity White Paper CSWP*, 39, 2025.

[BCG+17] Zvika Brakerski, Nishanth Chandran, Vipul Goyal, Aayush Jain, Amit Sahai, and Gil Segev. Hierarchical functional encryption. In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[BF01] Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In *Annual international cryptology conference*, pages 213–229. Springer, 2001.

[BLM+24] Pedro Branco, Russell W. F. Lai, Monosij Maitra, Giulio Malavolta, Ahmadreza Rahimi, and Ivy K. Y. Woo. Traitor tracing without trusted authority from registered functional encryption. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology – ASIACRYPT 2024, Part III*, volume 15486 of *Lecture Notes in Computer Science*, pages 33–66, Kolkata, India, December 9–13, 2024. Springer, Singapore, Singapore.

[BP26] Luís TAN Brandão and Rene Peralta. Nist first call for multi-party threshold schemes. Technical report, National Institute of Standards and Technology, 2026.

[BS95] Dan Boneh and James Shaw. Collusion-secure fingerprinting for digital data (extended abstract). In Don Coppersmith, editor, *Advances in Cryptology – CRYPTO'95*, volume 963 of *Lecture Notes in Computer Science*, pages 452–465, Santa Barbara, CA, USA, August 27–31, 1995. Springer Berlin Heidelberg, Germany.

[BSW06] Dan Boneh, Amit Sahai, and Brent Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 573–592, St. Petersburg, Russia, May 28 – June 1, 2006. Springer Berlin Heidelberg, Germany.

[BTVW17] Zvika Brakerski, Rotem Tsabary, Vinod Vaikuntanathan, and Hoeteck Wee. Private constrained PRFs (and more) from LWE. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017: 15th Theory of Cryptography Conference, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 264–302, Baltimore, MD, USA, November 12–15, 2017. Springer, Cham, Switzerland.

[BW06] Dan Boneh and Brent Waters. A fully collusion resistant broadcast, trace, and revoke system. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006: 13th Conference on Computer and Communications Security*, pages 211–220, Alexandria, Virginia, USA, October 30 – November 3, 2006. ACM Press.

[BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In *34rd Annual International Cryptology Conference, CRYPTO 2014*, pages 480–499. Springer Verlag, 2014.

[CC09] Melissa Chase and Sherman S. M. Chow. Improving privacy and security in multi-authority attribute-based encryption. In *ACM Conference on Computer and Communications Security*, pages 121–130, 2009.

[CC17] Ran Canetti and Yilei Chen. Constraint-hiding constrained PRFs for NC^1 from LWE. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part I*, volume 10210 of *Lecture Notes in Computer Science*, pages 446–476, Paris, France, April 30 – May 4, 2017. Springer, Cham, Switzerland.

[CFN94] Benny Chor, Amos Fiat, and Moni Naor. Tracing traitors. In Yvo Desmedt, editor, *Advances in Cryptology – CRYPTO'94*, volume 839 of *Lecture Notes in Computer Science*, pages 257–270, Santa Barbara, CA, USA, August 21–25, 1994. Springer Berlin Heidelberg, Germany.

[Cha07] Melissa Chase. Multi-authority attribute based encryption. In *Theory of Cryptography: 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007. Proceedings 4*, pages 515–534. Springer, 2007.

[CMMS25] Anirban Chakrabarti, Monosij Maitra, Arup Mondal, and Kushaz Sehgal. Silent threshold traitor tracing & enhancing mempool privacy. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, CCS '25, page 1769–1783, New York, NY, USA, 2025. Association for Computing Machinery.

[CVW+18a] Yilei Chen, Vinod Vaikuntanathan, Brent Waters, Hoeteck Wee, and Daniel Wichs. Traitor-tracing from lwe made simple and attribute-based. In *Theory of Cryptography: 16th International Conference, TCC 2018, Panaji, India, November 11–14, 2018, Proceedings, Part II 16*, pages 341–369. Springer, 2018.

[CVW18b] Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. Ggh15 beyond permutation branching programs: proofs, attacks, and candidates. In *Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part II 38*, pages 577–607. Springer, 2018.

[DKW21] Pratish Datta, Ilan Komargodski, and Brent Waters. Decentralized multi-authority abe for dnf s from lwe. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 177–209. Springer, 2021.

[DNR+09] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil P. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 381–390, Bethesda, MD, USA, May 31 – June 2, 2009. ACM Press.

[DP23] Pratish Datta and Tapas Pal. Decentralized multi-authority attribute-based inner-product fe: Large universe and unbounded. In *IACR International Conference on Public-Key Cryptography*, pages 587–621. Springer, 2023.

[GGL24] Rachit Garg, Rishab Goyal, and George Lu. Dynamic collusion functional encryption and multi-authority attribute-based encryption. In *IACR International Conference on Public-Key Cryptography*, pages 69–104. Springer, 2024.

[GKM+19] Rishab Goyal, Sam Kim, Nathan Manohar, Brent Waters, and David J. Wu. Watermarking public-key cryptographic primitives. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 367–398, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Cham, Switzerland.

[GKRW18] Rishab Goyal, Venkata Koppula, Andrew Russell, and Brent Waters. Risky traitor tracing and new differential privacy negative results. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 467–497, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Cham, Switzerland.

[GKW17] Rishab Goyal, Venkata Koppula, and Brent Waters. Lockable obfuscation. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017*, pages 612–621, 2017.

[GKW18] Rishab Goyal, Venkata Koppula, and Brent Waters. Collusion resistant traitor tracing from learning with errors. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 660–670, 2018.

[GKW19] Rishab Goyal, Venkata Koppula, and Brent Waters. New approaches to traitor tracing with embedded identities. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019: 17th Theory of Cryptography Conference, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 149–179, Nuremberg, Germany, December 1–5, 2019. Springer, Cham, Switzerland.

[GLW23] Junqing Gong, Ji Luo, and Hoeteck Wee. Traitor tracing with $N^{1/3}$-size ciphertexts and $O(1)$-size keys from k-Lin. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part III*, volume 14006 of *Lecture Notes in Computer Science*, pages 637–668, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

[GQWW19] Rishab Goyal, Willy Quach, Brent Waters, and Daniel Wichs. Broadcast and trace with N^{ϵ} ciphertext size from standard assumptions. In Alexandra Boldyreva and

Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 826–855, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Cham, Switzerland.

[GY24] Rishab Goyal and Saikumar Yadugiri. Multi-authority functional encryption with bounded collusions from standard assumptions. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024: 22nd Theory of Cryptography Conference, Part III*, volume 15366 of *Lecture Notes in Computer Science*, pages 3–30, Milan, Italy, December 2–6, 2024. Springer, Cham, Switzerland.

[GY25] Rishab Goyal and Saikumar Yadugiri. Decentralized multi-authority functional encryption with malicious authorities from standard assumptions. Cryptology ePrint Archive, Paper 2025/412, 2025.

[Luo24] Ji Luo. Ad hoc broadcast, trace, and revoke: Plus time-space trade-offs for attribute-based encryption. *IACR Communications in Cryptology (CiC)*, 1(2):15, 2024.

[LW11] Allison B. Lewko and Brent Waters. Decentralizing attribute-based encryption. In *EUROCRYPT*, pages 568–588, 2011.

[NNL01] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62, Santa Barbara, CA, USA, August 19–23, 2001. Springer Berlin Heidelberg, Germany.

[NWZ16] Ryo Nishimaki, Daniel Wichs, and Mark Zhandry. Anonymous traitor tracing: How to embed arbitrary information in a key. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 388–419, Vienna, Austria, May 8–12, 2016. Springer Berlin Heidelberg, Germany.

[Rog15] Phillip Rogaway. The moral character of cryptographic work. Cryptology ePrint Archive, Paper 2015/1162, 2015.

[Sha85] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology: Proceedings of CRYPTO 84 4*, pages 47–53. Springer, 1985.

[WW24] Brent Waters and Daniel Wichs. Adaptively secure attribute-based encryption from witness encryption. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024: 22nd Theory of Cryptography Conference, Part III*, volume 15366 of *Lecture Notes in Computer Science*, pages 65–90, Milan, Italy, December 2–6, 2024. Springer, Cham, Switzerland.

[WWW22] Brent Waters, Hoeteck Wee, and David J Wu. Multi-authority abe from lattices without random oracles. In *Theory of Cryptography Conference*, pages 651–679. Springer, 2022.

[WZ17] Daniel Wichs and Giorgos Zirdelis. Obfuscating compute-and-compare programs under LWE. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017*, pages 600–611, 2017.

[Zha20] Mark Zhandry. New techniques for traitor tracing: Size $N^{1/3}$ and more from pairings. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part I*, volume 12170 of *Lecture Notes in Computer Science*, pages 652–682, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Cham, Switzerland.

[Zha21] Mark Zhandry. White box traitor tracing. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part IV*, volume 12828 of *Lecture Notes in Computer Science*, pages 303–333, Virtual Event, August 16–20, 2021. Springer, Cham, Switzerland.

[Zha25] Mark Zhandry. Optimal traitor tracing from pairings. In Serge Fehr and Pierre-Alain Fouque, editors, *Advances in Cryptology – EUROCRYPT 2025, Part III*, volume 15603 of *Lecture Notes in Computer Science*, pages 305–335, Madrid, Spain, May 4–8, 2025. Springer, Cham, Switzerland.

A Multi-Authority Traitor Tracing from Multi-Authority PLBE

A.1 Multi-Authority Decoder Private Linear Broadcast Encryption

Syntax. A Multi-Authority Decoder Private Linear Broadcast Encryption (MA-D-PLBE) scheme for message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_\lambda$ consists of the following polynomial time algorithms.

$\mathsf{GSetup}(1^\lambda, 1^K, 1^N) \to \mathsf{crs}$ The randomized global setup algorithm takes as input security parameter λ, number of authorities K, number of users N, and outputs common reference string crs, which is assumed to be an input of all the following functions.

$\mathsf{ASetup}(i) \to (\mathsf{mpk}_i, \mathsf{msk}_i)$ The randomized authority setup algorithm takes as input an authority index $i \in [K]$, and outputs associated master secret key msk_i and master public key mpk_i.

$\mathsf{KGen}(i, \mathsf{msk}_i, j) \to \mathsf{sk}_{i,j}$ The randomized key generation algorithms takes as input an authority index i along with an associated master secret key msk_i. Then it uses the inputted user index $j \in [N]$ to generate a user secret key associated with authority i and user j called $\mathsf{sk}_{i,j}$.

$\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m) \to \mathsf{ct}$ The randomized encryption algorithm takes as input a master public key mpk_i from each authority index, and a message m. It returns a ciphertext ct.

$\mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, j, m) \to \mathsf{ct}$ The randomized secret key encryption algorithm takes as input all authority master secret keys msk_i the message m, and a user index $j \in [0, ..., N]$. Returning a ciphertext ct.

$\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct}) \to m/\perp$ The deterministic decryption algorithm takes a ciphertext ct and a set of secret keys $\mathsf{sk}_{i,j}$ generated by each authority index under the same user index j. Then it either returns the message m or $\perp$.

Definition A.1. A Multi-Authority Decoder Private Linear Broadcast Encryption scheme (GSetup, ASetup, KGen, Enc, $\mathsf{EncIndex}$, Dec) must satisfy the following properties.

Correctness Identical to PLBE Correctness on page 16

Normal Hiding Security For any admissible PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{rcl} & & (1^K, 1^N) \leftarrow \mathcal{A} \\ & & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N) \\ & & \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ D(\mathsf{ct}_b) = b & : & (\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}) \rightarrow \mathcal{A} \\ & & (D, m) \leftarrow \mathcal{A}^{\mathsf{KGen}(\cdot,\mathsf{msk}_{(\cdot)},\cdot),\mathsf{Corr}(\cdot)} \\ & & b \leftarrow \{0,1\} \\ & & \mathsf{ct}_0 \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m) \\ & & \mathsf{ct}_1 \leftarrow \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, 0, m) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

The corruption oracle Corr takes an input authority index $i \in [K]$ and responds with msk_i. $\mathcal{A}$ cannot query Corr for every authority index.
The KGen oracle takes authority index $i \in [K]$ and user index $j \in [N]$ and responds with a secret key generated under i and j.
There are no restrictions on the use of the KGen oracle.

Index Hiding Security For any admissible PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{rcl} & & (1^K, 1^N, j) \leftarrow \mathcal{A} \\ & & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N) \\ & & \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ D(\mathsf{ct}) = b & : & (\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}) \rightarrow \mathcal{A} \\ & & (D, m) \leftarrow \mathcal{A}^{\mathsf{KGen}(\cdot,\mathsf{msk}_{(\cdot)},\cdot),\mathsf{Corr}(\cdot)} \\ & & b \leftarrow \{0,1\} \\ & & \mathsf{ct} \leftarrow \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, j-b, m) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

The corruption oracle Corr takes an input authority index $i \in [K]$ and responds with msk_i. $\mathcal{A}$ cannot query Corr for every authority index.
The KGen oracle takes authority index $i \in [K]$ and user index $j \in [N]$ and responds with a secret key generated under i and j.
Let U be the set of authority indexes queried by the Corr oracle, $\mathcal{A}$ cannot query the KGen oracle for every $\{(i,j)\}_{i\in[K]\setminus U}$.

Message Hiding Security For any PPT stateful adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{rcl} & & (1^K, 1^N) \leftarrow \mathcal{A} \\ & & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^K, 1^N) \\ & & \forall\ i \in [K], (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{ASetup}(i) \\ D(\mathsf{ct}) = b & : & (\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}) \rightarrow \mathcal{A} \\ & & (D, m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{KGen}(\cdot,\mathsf{msk}_{(\cdot)},\cdot),\mathsf{Corr}(\cdot)} \\ & & b \leftarrow \{0,1\} \\ & & \mathsf{ct} \leftarrow \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, 1^N, m_b) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

The corruption oracle Corr takes an input authority index $i \in [K]$ and responds with msk_i. $\mathcal{A}$ cannot query Corr for every authority index.
The KGen oracle takes authority index $i \in [K]$ and user index $j \in [N]$ and responds with a secret key generated under i and j.

There are no restrictions on the use of the KGen oracle.

A.2 Multi-Authority Traitor Tracing from MA-D-PLBE

To build Multi-Authority Traitor Tracing from Multi-Authority Decoder PLBE we follow the original Traitor Tracing construction from [GKW18]. The only challenging aspect of this construction is building the Trace algorithm. The idea is to take a decoder box D, and iterate through all possible user identities $j \in [N]$. The MA-D-PLBE algorithm $\mathsf{EncIndex}$ creates a ciphertext that is impossible to decrypt for all users with identity less than or equal to j. The Index Hiding Security property tells us that a decoder box can only distinguish between two consecutive $\mathsf{EncIndex}$ encryptions if it has access to user j, so we create a large amount of ciphertexts (W) to get an approximation for the decoder box's decryption rate at each user index ($\hat{p}_j$). If we see a large enough jump in the decryption rate between two consecutive index encryptions ($\hat{\mu}_j$) then we conclude that the decoder box must have access to that user index.

$\mathsf{GSetup}(1^\lambda, 1^K, 1^N) \to \mathsf{crs}$ Perform PLBE's global setup algorithm.
$\mathsf{crs} \leftarrow \mathsf{PLBE.GSetup}(1^\lambda, 1^K, 1^N)$

$\mathsf{ASetup}(i) \to (\mathsf{mpk}_i, \mathsf{msk}_i)$ Perform PLBE's authority setup algorithm.
$(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{PLBE.ASetup}(i)$

$\mathsf{KGen}(i, \mathsf{msk}_i, j) \to \mathsf{sk}_{i,j}$ Perform PLBE's key generation algorithm.
$\mathsf{sk}_{i,j} \leftarrow \mathsf{PLBE.KGen}(i, \mathsf{msk}_i, j)$

$\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m) \to ct$ Perform PLBE's public key encryption algorithm.
$ct \leftarrow \mathsf{PLBE.Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m)$

$\mathsf{Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct}) \to (m/\perp)$ Perform PLBE's decryption algorithm.
$(m/\perp) \leftarrow \mathsf{PLBE.Dec}(\{\mathsf{sk}_{i,j}\}_{i\in[K]}, \mathsf{ct})$

$\mathsf{Trace}^D(\{\mathsf{msk}_i\}_{i\in[K]}, 1^y, m_0, m_1) \to T$ Iterate over all user indexes, for each index we sample W secret key encryptions over index j and test the decoder box's ability to distinguish between m_0 and m_1 at each step. If we notice a large enough jump in decryption probability between two indexes $j-1$ and j then we know that the decoder box must contain a valid set of secret

keys for user index j.

$W = \lambda N^2 y^2$

$\mathsf{For}\ j \in [0, \ldots, N]$

$\quad \mathsf{count}_j = 0$

$\quad \mathsf{For}\ k \in [0, \ldots, W]:$

$\quad\quad b_{j,k} \leftarrow \{0,1\}$

$\quad\quad \mathsf{ct}_{j,k} \leftarrow \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, j, m_b)$

$\quad\quad \mathsf{If}(D(\mathsf{ct}_{j,k}) = b_{j,k}):$

$\quad\quad\quad \mathsf{Then}:\ \mathsf{count}_j = \mathsf{count}_j + 1$

$\hat{p}_j = \mathsf{count}_j / W$

$\hat{\mu}_j = \hat{p}_{j-1} - \hat{p}_j$

$T = \{\ j \mid \hat{\mu}_j \geq \frac{1}{4Ny}\}$

Theorem A.2. If PLBE is a valid Multi-Authority Decoder PLBE scheme, then the construction on page 45 is a valid Multi-Authority Traitor Tracing scheme.

Correctness Identical to the first correctness property of Decoder PLBE on page 43.

Semantic Security An admissible adversary $\mathcal{A}$ in the Traitor Tracing Semantic Security game cannot query all $\{(i,j)\}_{i\in[K]\setminus U}$ for any particular user index j. This means that $\mathcal{A}$'s key generation queries (when reduced to the Decoder PLBE challenger) will be admissible in the Decoder PLBE Normal Hiding Security game, Message Hiding Security game, and notably in Index Hiding Security game irrespective of the chosen challenge index.

First we create a hybrid experiment $\mathsf{Hybrid}^{\mathcal{A}}(b)$, identical to the Traitor Tracing semantic security experiment. Except if the challenger receives (m_0, m_1) they always return $\mathsf{ct} \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m_b)$.

Next we define $\mathsf{Hybrid}^{\mathcal{A}}(b, j)$ which is identical to the previous except the challenger returns $\mathsf{ct} \leftarrow \mathsf{EncIndex}(\{\mathsf{msk}_i\}_{i\in[K]}, j, m_b)$

Normal Hiding Security gives us $\mathsf{Hybrid}^{\mathcal{A}}(b) \overset{c}{\approx} \mathsf{Hybrid}^{\mathcal{A}}(b, 0)$, Index Hiding Security gives us $\mathsf{Hybrid}^{\mathcal{A}}(b, j) \overset{c}{\approx} \mathsf{Hybrid}^{\mathcal{A}}(b, j+1)$, and Message Hiding Security gives us $\mathsf{Hybrid}^{\mathcal{A}}(0, N) \overset{c}{\approx} \mathsf{Hybrid}^{\mathcal{A}}(1, N)$.

If we combine all these statements the result is $\mathsf{Hybrid}^{\mathcal{A}}(0) \overset{c}{\approx} \mathsf{Hybrid}^{\mathcal{A}}(1)$. Which means $\mathcal{A}$ cannot distinguish between $\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m_0)$ and $\mathsf{Enc}(\{\mathsf{mpk}_i\}_{i\in[K]}, m_1)$ with non-negligible probability.

Secure Traitor Tracing

Lemma A.3. If PLBE satisies the Index-Hiding Security property of MA-D-PLBE, then given an adversary $\mathcal{A}$ for the Secure Traitor Tracing Experiment, the probability of the FalseTrace is negligible.

$$\Pr\left[\ T \not\subset S \quad : \quad (D, S, T) \leftarrow \mathsf{Tr}^{\mathcal{A}}(1^\lambda, 1^y)\ \right] \leq \mathsf{negl}(\lambda)$$

Proof. The following probabilities are taken over the randomness of Enc and EncIndex in the construction above.

Let $p_j^D = \Pr\left[\ D(\mathsf{ct}) = b \quad : \quad \begin{array}{l} b \leftarrow \{0,1\} \\ \mathsf{EncIndex}(\{\mathsf{msk}_i\}, j, m_b) \rightarrow \mathsf{ct} \end{array}\ \right]$

Let $p_{nrml}^D = \Pr\left[\ D(\mathsf{ct}) = b \quad : \quad \begin{array}{l} b \leftarrow \{0,1\} \\ \mathsf{Enc}(\{\mathsf{mpk}_i\}, m_b) \rightarrow \mathsf{ct} \end{array}\ \right] = \mathsf{Adv}^D$

Let $\mu_j = p_{j-1}^D - p_j^D$

The following probabilities are taken over the randomness of the tracing experiment using the construction above $(D, S, T, \{\hat{p}_j, \hat{\mu}_j, \{\mathsf{ct}_{j,k}, b_{j,k}\}_{k\in[W]}\}_{j\in[N]}) \leftarrow \mathsf{Tr}^{\mathcal{A}}(1^\lambda, 1^y)$. Using union bound we can rearrange the lemma above.

$$\Pr[T \not\subset S] \leq \sum_{j\notin S} \Pr[j \in T] = \sum_{j\notin S} \Pr\left[\hat{\mu}_j \geq \frac{1}{4Ny}\right]$$

$$\Pr[T \not\subset S] \leq \sum_{j\notin S} \Pr\left[\hat{\mu}_j \geq \frac{1}{4Ny} \wedge \mu_j > \frac{1}{8Ny}\right] + \sum_{j\notin S} \Pr\left[\hat{\mu}_j \geq \frac{1}{4Ny} \wedge \mu_j \leq \frac{1}{8Ny}\right]$$

$$\Pr[T \not\subset S] \leq \sum_{j\notin S} \Pr\left[\mu_j > \frac{1}{8Ny}\right] + \sum_{j\notin S} \Pr\left[\hat{\mu}_j \geq \frac{1}{4Ny} \wedge \mu_j \leq \frac{1}{8Ny}\right]$$

Lemma A.4. If PLBE satisfies the Index Hiding Security property of MA-D-PLBE then:

$$\Pr\left[\mu_{j^*} \geq \frac{1}{8Ny} \ \mid\ j^* \notin S\right] \leq \mathsf{negl}(\lambda)$$

Proof. Suppose there exists an adversary such that for some j^*, $\Pr\left[\mu_{j^*} \geq \frac{1}{8Ny} \ \mid\ j^* \notin S\right] \geq \delta(\lambda)$ where δ is some non-negligible function. Then we can break the index-hiding security of our Decoder PLBE scheme. First, the Tracing Experiment adversary sends us $(1^K, 1^N)$ we choose a random $j' \in [N]$ and give $(1^K, 1^N, j')$ to the index hiding Decoder PLBE challenger. Then the adversary makes key generation requests for (i, j) and authority corruptions requests for i, we forward these to the PLBE challenger who responds with the corresponding secret keys $\mathsf{sk}_{i,j}$ or authority master secret keys msk_i. If the adversary requests enough master secret keys $i \in U$ and secret keys $(i, j) \in Q$ such that they collect all non-corrupted secret keys for user j' then we abort. If not, the adversary will eventually give us (D, m_0, m_1) and we will flip a bit $b' \leftarrow \{0,1\}$. Create a new decoder box $D'(\mathsf{ct}) = (D(\mathsf{ct}) = b')$. Then, send the challenger $(D', m_{b'})$.

Let $p_{j,b}^D = \Pr\left[\ D(\mathsf{ct}_b) = b \quad : \quad \mathsf{ct}_b = \mathsf{EncIndex}(\{\mathsf{msk}_i\}, j, m_b)\ \right]$

$$\Pr\left[D'(\mathsf{EncIndex}(\{\mathsf{msk}_i\}, j'-1, m_{b'}) = 1\right] = p_{j'-1,b'}^D$$

$$\Pr\left[D'(\mathsf{EncIndex}(\{\mathsf{msk}_i\}, j', m_{b'}) = 1\right] = p_{j',b'}^D$$

$$\mathsf{Adv} = p_{j'-1,b'}^D - p_{j',b'}^D$$

By our initial assumption, there must exist some non-negligible function δ such that.

$$\Pr\left[\mu_{j^*} \geq \frac{1}{8Ny}\right] \geq \delta(\lambda)$$

$$\Pr\left[p^D_{j^*-1} - p^D_{j^*} \geq \frac{1}{8Ny}\right] \geq \delta(\lambda)$$

$$\Pr\left[\frac{p^D_{j^*-1,0} + p^D_{j^*-1,1}}{2} - \frac{p^D_{j^*,0} + p^D_{j^*,1}}{2} \geq \frac{1}{8Ny}\right] \geq \delta(\lambda)$$

$$\Pr\left[(p^D_{j^*-1,0} - p^D_{j^*,0}) + (p^D_{j^*-1,1} - p^D_{j^*,1}) \geq \frac{1}{4Ny}\right] \geq \delta(\lambda)$$

So there must exist a bit b^* such that:

$$\Pr\left[p^D_{j^*-1,b^*} - p^D_{j^*,b^*} \geq \frac{1}{8Ny}\right] \geq \delta(\lambda)$$

If the reduction algorithm correctly choses $j' = j^*$ and $b' = b^*$ which happens with probability $\frac{1}{2N}$. We can conclude

$$\Pr\left[p^D_{j'-1,b'} - p^D_{j',b'} \geq \frac{1}{8Ny}\right] \geq \frac{\delta(\lambda)}{2N}$$

$$\Pr\left[\mathsf{Adv} \geq \frac{1}{8Ny}\right] \geq \frac{\delta(\lambda)}{2N}$$

So the total advantage of the reduction algorithm taken over the randomness of our reduction will be.

$$\mathsf{Adv} \geq \frac{\delta(\lambda)}{16N^2y}$$

This is a contradiction of the Index Hiding Security property of our Decoder PLBE scheme. So by contradiction we get that

$$\Pr\left[\mu_{j^*} \geq \frac{1}{8Ny} \;\middle|\; j^* \notin S\right] \leq \mathsf{negl}(\lambda)$$

□

Lemma A.5.

$$\Pr\left[\hat{\mu}_j \geq \frac{1}{4Ny} \wedge \mu_j \leq \frac{1}{8Ny}\right] \leq \mathsf{negl}(\lambda)$$

Proof. First, we bound the original probability by a slightly modified one.

$$\Pr\left[\hat{\mu}_j \geq \frac{1}{4Ny} \wedge \mu_j \leq \frac{1}{8Ny}\right] \leq \Pr\left[\hat{\mu_j} \geq \frac{1}{4Ny} \;\middle|\; \mu_j \leq \frac{1}{8Ny}\right]$$

Next, for all $j \in [N], k \in [W]$ let $X_{j,k}$ be a random variable representing the event that $D(\mathsf{ct}_{j,k}) = b_{j,k}$. Then let $Z_{j,k}$ be the random variable representing $X_{j-1,k} - X_{j,k}$. Then $\hat{\mu}_j = \sum_{k=0}^{W} Z_{j,k}/W$ and $\mu_j = E(Z_{j,k})$. Notice that $-1 \leq Z_{i,j} \leq 1$. Using the Chernoff-Hoeffding bound for some $j \notin S$

where $\mu_j \leq \frac{1}{8Ny}$ we get.

$$\Pr\left[\frac{1}{W}(\sum_{k=0}^{W} Z_{j,k} - \sum_{k=0}^{W} E(Z_{j,k})) \geq \frac{1}{8Ny}\right] \leq e^{-2(\frac{1}{8Ny})^2/(\sum_0^W 1-(-1))}$$

$$\Pr\left[\hat{\mu}_j \geq \frac{1}{8Ny} + \mu_j\right] \leq e^{-\frac{W}{64N^2y^2}}$$

$$\Pr\left[\hat{\mu}_j \geq \frac{1}{8Ny} + \frac{1}{8Ny}\right] \leq e^{-\frac{W}{64N^2y^2}}$$

$$\Pr\left[\hat{\mu}_j \geq \frac{1}{4Ny}\right] \leq e^{-\frac{\lambda}{64}} = e^{-\omega(\mathsf{polylog}(\lambda))}$$

$$\Pr\left[\hat{\mu}_j \geq \frac{1}{4Ny}\right] \leq \mathsf{negl}(\lambda)$$

□

Putting together the Lemma A.4 and Lemma A.5 with our previous work on Lemma A.3, we can finally conclude that our construction bounds the FalseTrace event.

$$\Pr[T \not\subset S] \leq \sum_{j \notin S} \mathsf{negl}(\lambda) + \mathsf{negl}(\lambda) = (N+1)\mathsf{negl}(\lambda) = \mathsf{negl}(\lambda)$$

□

Lemma A.6. $$\Pr[T \neq \emptyset] \geq \Pr[\mathsf{GoodDecoder}] - \mathsf{negl}(\lambda)$$

Proof. In the GoodDecoder event we know that $p^D_{nrml} \geq \frac{1}{2} + \frac{1}{y}$ and by Message Hiding Security we know that $p^D_N \leq \frac{1}{2} + \mathsf{negl}(\lambda)$. Then by Normal Hiding Security we know that $|p^D_0 - p^D_{nrml}| \leq \mathsf{negl}(\lambda)$

$$\sum_{j=1}^{N} p^D_{j-1} - p^D_j = p^D_0 - p^D_N \geq p^D_{nrml} - \mathsf{negl}(\lambda) - p^D_N \geq \frac{1}{y} - \mathsf{negl}(\lambda)$$

So there must exist some $j \in [N]$ such that

$$\mu_j = p^D_{j-1} - p^D_j \geq \frac{1}{N}(\frac{1}{y} - \mathsf{negl}(\lambda)) \geq \frac{1}{2Ny}$$

Using the Chernoff-Hoeffding bound we have that:

$$\Pr\left[\frac{1}{W}(\sum_{k=0}^{W} Z_{j,k} - \sum_{k=0}^{W} E(Z_{j,k})) \leq -\frac{1}{4Ny}\right] \leq e^{-2(\frac{W}{8Ny})^2/2W}$$

$$\Pr\left[\hat{\mu}_j - \mu_j \leq -\frac{1}{4Ny}\right] \leq \Pr\left[\hat{\mu}_j - \frac{1}{2Ny} \leq -\frac{1}{4Ny}\right] \leq e^{-\frac{\lambda}{64}}$$

$$\Pr\left[\hat{\mu}_j \leq \frac{1}{4Ny}\right] \leq e^{-\frac{\lambda}{64}}$$

$$\Pr[j \notin T] \leq \mathsf{negl}(\lambda)$$

So now we have that given the GoodDecoder event there exists a $j^* \in S$ such that

$\Pr[j^* \in T] > 1 - negl(\lambda)$.

$$\Pr[T \neq \emptyset] \geq \Pr[j^* \in T] = \Pr\left[\ j^* \in T \quad : \quad \mathsf{GoodDecoder}\ \right] * \Pr[\mathsf{GoodDecoder}]$$

$$\Pr[T \neq \emptyset] \geq (1 - \mathsf{negl}(\lambda)) \Pr[\mathsf{GoodDecoder}]$$

$$\Pr[T \neq \emptyset] \geq \Pr[\mathsf{GoodDecoder}] - \mathsf{negl}(\lambda) * \Pr[\mathsf{GoodDecoder}]$$

$$\Pr[T \neq \emptyset] \geq \Pr[\mathsf{GoodDecoder}] - \mathsf{negl}(\lambda)$$

□

A.3 Multi-Authority Decoder PLBE from Multi-Authority PLBE

Theorem A.7. If $\mathsf{PLBE} = (\mathsf{GSetup}, \mathsf{ASetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{EncIndex}, \mathsf{Dec})$ is a valid Multi-Authority PLBE scheme then it is also a valid Multi-Authority Decoder-PLBE scheme.

Lemma A.8. If PLBE satisfies the MA-PLBE Normal Hiding Security property then PLBE must satisfy the MA-D-PLBE Normal Hiding Security property.

Proof. Suppose for the sake of contradiction that PLBE does not satisfy the MA-D-PLBE Normal Hiding Security property. Then there must exist an adversary $\mathcal{A}$ with non-negligible advantage in the MA-D-PLBE Normal Hiding Security game. Let challenger $\mathcal{C}$ be for the MA-PLBE Normal Hiding Security game, then reduction $\mathcal{R}$ is as follows:

Setup $(1^K, 1^N)$ $\mathcal{R}$ guesses an honest authority $h \in [K]$, and sets $U_h = [K] \backslash \{h\}$, then $\mathcal{R}$ sends the challenger $(1^K, 1^N, U_h)$. The challenger $\mathcal{C}$ responds with $(\mathsf{crs}, \{\mathsf{mpk}_i\}_{i \in [K]}, \{\mathsf{msk}_i\}_{i \in U_h}$, and $\mathcal{R}$ forwards $(\mathsf{crs}, \{\mathsf{mpk}_i\}_{i \in [K]})$ to $\mathcal{A}$.

Query $\mathsf{Corr}(i)$ If $i \notin U_h$ then abort $\mathcal{R}$. Otherwise, $\mathcal{R}$ responds to $\mathcal{A}$ with msk_i.

Query $\mathsf{KGen}(i, j)$ $\mathcal{R}$ queries (i, j) to $\mathcal{C}$'s key generation oracle. $\mathcal{C}$ responds with $\mathsf{sk}_{i,j}$ and $\mathcal{R}$ forwards this to $\mathcal{A}$.

Challenge Phase (D, m) First, $\mathcal{R}$ makes it's 1-Bounded encryption query to $\mathcal{C}$ by sending m (with the assumed user index of 0). $\mathcal{C}$ responds with ct_1^*. Next, $\mathcal{R}$ encrypts m in the public key mode $\mathsf{ct}_0^* \leftarrow \mathsf{Enc}(\{\mathsf{mpk}_i\}_{i \in [K]}$. Then $\mathcal{R}$ makes it's challenge request to $\mathcal{C}$ by sending m. $\mathcal{C}$ responds with ct_b. Finally $\mathcal{R}$ flips a bit $b^* \leftarrow \{0, 1\}$ and tests the decoder box on two ciphertexts $b'^* \leftarrow D(\mathsf{ct}_{b^*}^*)$, $b' \leftarrow D(\mathsf{ct}_b)$. If $b'^* = b'$ then $\mathcal{R}$ returns b', if not, $\mathcal{R}$ returns $1 - b'$.

Given an adversary $\mathcal{A}$ with non-negligible advantage ϵ against the MA-D-PLBE NHS game.

$$\Pr[b \leftarrow \mathcal{R}] = \Pr[D(\mathsf{ct}_{b^*}^*) = b^*] * \Pr[D(\mathsf{ct}_b) = b] + \Pr[D(\mathsf{ct}_{b^*}^*) \neq b^*] * \Pr[D(\mathsf{ct}_b) \neq b]$$

$$\Pr\left[b' = b\right] = (\frac{1}{2} + \epsilon)(\frac{1}{2} + \epsilon) + (\frac{1}{2} - \epsilon)(\frac{1}{2} - \epsilon)$$

$$\Pr\left[b' = b\right] = \frac{1}{2} + 2\epsilon^2$$

So $\mathcal{R}$ will have advantage $\frac{1}{2} + 2\epsilon^2$ if the program doesn't halt early. In the worst case $\mathcal{A}$ can only query $K - 1$ authorities for corruption. This means there always exists an authority index $h^* \in [K]$ which remains honest. The event that $h^* = h$ happens with probability $\frac{1}{K}$, so $\mathcal{R}$ has non-negligible advantage of $\frac{1}{2} + \frac{2\epsilon^2}{K}$ in the MA-PLBE NHS game. □

Lemma A.9. If PLBE satisfies the MA-PLBE Index Hiding Security property then PLBE must satisfy the MA-D-PLBE Index Hiding Security property.

Proof. Suppose for the sake of contradiction that PLBE does not satisfy the MA-D-PLBE Index Hiding Security property. Then there must exist an adversary $\mathcal{A}$ with non-negligible advantage in the MA-D-PLBE Index Hiding Security game. Let challenger $\mathcal{C}$ be for the MA-PLBE Index Hiding Security game, then reduction $\mathcal{R}$ is as follows:

Setup $(1^K, 1^N, j)$ $\mathcal{R}$ guesses a honest authority $h \in [K]$, and sets $U_h = [K] \backslash \{t\}$, then $\mathcal{R}$ sends the challenger $(1^K, 1^N, U_h)$. The challenger $\mathcal{C}$ responds with $(\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}, \{\mathsf{msk}_i\}_{i\in U_h}$, and $\mathcal{R}$ forwards $(\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]})$ to $\mathcal{A}$.

Query $\mathsf{Corr}(i)$ If $i \notin U_h$ then abort $\mathcal{R}$. Otherwise, $\mathcal{R}$ responds to $\mathcal{A}$ with msk_i.

Query $\mathsf{KGen}(i, j)$ $\mathcal{R}$ queries (i, j) to $\mathcal{C}$'s key generation oracle. $\mathcal{C}$ responds with $\mathsf{sk}_{i,j}$ and $\mathcal{R}$ forwards this to $\mathcal{A}$.

Challenge Phase (D, m) First, $\mathcal{R}$ samples $b^* \leftarrow \{0, 1\}$, and makes it's 1-Bounded encryption query to $\mathcal{C}$ by sending $(m, j - b^*)$. $\mathcal{C}$ responds with $\mathsf{ct}^*_{b^*}$. Then $\mathcal{R}$ makes it's challenge request to $\mathcal{C}$ by sending m. $\mathcal{C}$ responds with ct_b. Finally $\mathcal{R}$ tests the decoder box on two ciphertexts $b'^* \leftarrow D(\mathsf{ct}^*_{b^*})$, $b' \leftarrow D(\mathsf{ct}_b)$. If $b'^* = b'$ then $\mathcal{R}$ returns b', if not, $\mathcal{R}$ returns $1 - b'$.

Given an adversary $\mathcal{A}$ with non-negligible advantage ϵ against the MA-D-PLBE IHS game.

$$\Pr[b \leftarrow \mathcal{R}] = \Pr[D(\mathsf{ct}^*_{b^*}) = b^*] * \Pr[D(\mathsf{ct}_b) = b] + \Pr[D(\mathsf{ct}^*_{b^*}) \neq b^*] * \Pr[D(\mathsf{ct}_b) \neq b]$$

$$\Pr\left[b' = b\right] = (\frac{1}{2} + \epsilon)(\frac{1}{2} + \epsilon) + (\frac{1}{2} - \epsilon)(\frac{1}{2} - \epsilon)$$

$$\Pr\left[b' = b\right] = \frac{1}{2} + 2\epsilon^2$$

So $\mathcal{R}$ will have advantage $\frac{1}{2} + 2\epsilon^2$ if the program doesn't halt early. $\mathcal{R}$ will become an inadmissible adversary against $\mathcal{C}$ if it makes the KGen request for (t, j) or if it makes a corruption query for authority t. For $\mathcal{A}$ to be an admissible adversary in the IHS game, there must exist an authority index t^* that isn't corrupted and that (t^*, j) isn't queried by the KGen oracle. This means there always exists a valid authority index $h^* \in [K]$ which remains honest. The event that $h^* = h$ happens with probability $\frac{1}{K}$, so $\mathcal{R}$ has non-negligible advantage of $\frac{1}{2} + \frac{2\epsilon^2}{K}$ in the MA-PLBE IHS game. □

Lemma A.10. Message Hiding Security

Lemma A.11. If PLBE satisfies the MA-PLBE Message Hiding Security property then PLBE must satisfy the MA-D-PLBE Message Hiding Security property.

Proof. Suppose for the sake of contradiction that PLBE does not satisfy the MA-D-PLBE Message Hiding Security property. Then there must exist an adversary $\mathcal{A}$ with non-negligible advantage in the MA-D-PLBE Message Hiding Security game. Let challenger $\mathcal{C}$ be for the MA-PLBE Message Hiding Security game, then reduction $\mathcal{R}$ is as follows:

Setup $(1^K, 1^N)$ $\mathcal{R}$ guesses a honest authority $h \in [K]$, and sets $U_h = [K] \backslash \{h\}$, then $\mathcal{R}$ sends the challenger $(1^K, 1^N, U_h)$. The challenger $\mathcal{C}$ responds with $(\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]}, \{\mathsf{msk}_i\}_{i\in U_h}$, and $\mathcal{R}$ forwards $(\mathsf{crs}, \{\mathsf{mpk}_i\}_{i\in[K]})$ to $\mathcal{A}$.

Query $\mathsf{Corr}(i)$ If $i \notin U_h$ then abort $\mathcal{R}$. Otherwise, $\mathcal{R}$ responds to $\mathcal{A}$ with msk_i.

Query $\mathsf{KGen}(i, j)$ $\mathcal{R}$ queries (i, j) to $\mathcal{C}$'s key generation oracle. $\mathcal{C}$ responds with $\mathsf{sk}_{i,j}$ and $\mathcal{R}$ forwards this to $\mathcal{A}$.

Challenge Phase (D, m_0, m_1) First, $\mathcal{R}$ samples $b^* \leftarrow \{0, 1\}$, and makes it's 1-Bounded encryption query to $\mathcal{C}$ by sending (m_b, N). $\mathcal{C}$ responds with $\mathsf{ct}^*_{b^*}$. Then $\mathcal{R}$ makes it's challenge request to $\mathcal{C}$ by sending m. $\mathcal{C}$ responds with ct_b. Finally $\mathcal{R}$ tests the decoder box on the two ciphertexts $b'^* \leftarrow D(\mathsf{ct}^*_{b^*})$, $b' \leftarrow D(\mathsf{ct}_b)$. If $b'^* = b'$ then $\mathcal{R}$ returns b', if not, $\mathcal{R}$ returns $1 - b'$.

Given an adversary $\mathcal{A}$ with non-negligible advantage ϵ against the MA-D-PLBE MHS game.

$$\Pr[b \leftarrow \mathcal{R}] = \Pr[D(\mathsf{ct}^*_{b^*}) = b^*] * \Pr[D(\mathsf{ct}_b) = b] + \Pr[D(\mathsf{ct}^*_{b^*}) \neq b^*] * \Pr[D(\mathsf{ct}_b) \neq b]$$

$$\Pr\left[b' = b\right] = (\frac{1}{2} + \epsilon)(\frac{1}{2} + \epsilon) + (\frac{1}{2} - \epsilon)(\frac{1}{2} - \epsilon)$$

$$\Pr\left[b' = b\right] = \frac{1}{2} + 2\epsilon^2$$

So $\mathcal{R}$ will have advantage $\frac{1}{2} + 2\epsilon^2$ if the program doesn't halt early. In the worst case $\mathcal{A}$ can only query $K-1$ authorities for corruption. This means there always exists an authority index $h^* \in [K]$ which remains honest. The event that $h^* = h$ happens with probability $\frac{1}{K}$, so $\mathcal{R}$ has non-negligible advantage of $\frac{1}{2} + \frac{2\epsilon^2}{K}$ in the MA-PLBE MHS game. □