Decentralized Multi-Authority (Attribute-Based) Traitor Tracing

Pratish Datta
NTT Research
pratish.datta@ntt-research.com

Robert Schädlich
King's College London
robert.schaedlich@kcl.ac.uk

Erkan Tairi
University of California, Berkeley
erkan.tairi@berkeley.edu

Abstract

We initiate the study of multi-authority traitor tracing (MA-TT), a decentralized variant of traitor tracing in which tracing capabilities are distributed across multiple independent authorities rather than concentrated in a single trusted entity. Ciphertexts are associated with tracing policies over a collection of authorities, specifying which subsets of authorities are authorized to jointly accuse a user of contributing to a pirate decoder. This enables fine-grained control over tracing capabilities, prevents unilateral accusations, and reduces the surveillance risks inherent in centralized tracing systems. Our scheme naturally supports conjunction, disjunction, threshold, and more general monotone tracing policies. We further introduce multi-authority attribute-based traitor tracing (MA-AB-TT), which combines distributed tracing in MA-TT with decentralized access control from multi-authority attribute-based encryption (MA-ABE). Ciphertexts are therefore equipped with two orthogonal policies: an attribute policy governing decryption and a tracing policy governing which tracing authorities may jointly identify traitors.

Our main contribution is a construction of MA-AB-TT for arbitrary monotone access structures from the standard matrix decisional Diffie-Hellman (MDDH) assumption in prime-order pairing groups. Our construction achieves adaptive security under static corruption of authorities in the random oracle model. All system parameters are independent of the number of authorities and users in the system, while ciphertexts grow linearly with the size of the associated policies. Our framework also yields a publicly traceable variant, in which tracing can be performed using only the authorities' public keys, albeit with weaker asymptotic efficiency guarantees.

Technically, we extend the blueprint for pairing-based traitor tracing based on private linear broadcast encryption (PLBE) [Boneh et al., Eurocrypt 2006] to the multi-authority setting. The key technical ingredient is a new multi-authority PLBE construction, which we instantiate from slotted inner-product function encryption (sIPFE) [Lin and Luo, Eurocrypt 2020]. As an intermediate step, we provide a new attribute-based sIPFE scheme supporting arithmetic branching programs, which may be of independent interest.

Our techniques also yield a new modular construction of adaptively secure MA-ABE under static corruptions from sIPFE, improving on a prior construction by Ambrona and Gay [PKC 2023]. Finally, when specialized to a single authority, our framework gives the first ciphertext-policy attribute-based traitor tracing scheme with asymptotically optimal parameters and exponentially large user spaces.

Contents

1 Introduction **3**
- 1.1 Our Results . . . 5
- 1.2 Concurrent and Related Works . . . 7

2 Technical Overview **8**
- 2.1 A Generic Compiler for MA-TT . . . 8
- 2.2 Constructing Multi-Authority PLBE . . . 10
  - 2.2.1 A Modular MA-ABE Scheme from Slotted IPFE . . . 10
  - 2.2.2 Enabling Secret-Key Encryption . . . 11
- 2.3 Construction of Attribute-Based Slotted IPFE . . . 12
- 2.4 Full Traceability . . . 14
- 2.5 Towards Public Traceability . . . 16

3 Preliminaries **16**
- 3.1 Notational Conventions . . . 16
- 3.2 Pairing Groups and Hardness Assumptions . . . 17
- 3.3 Monotone Access Structures and Linear Secret Sharing Schemes . . . 17
- 3.4 Slotted Inner-Product Functional Encryption . . . 18
- 3.5 Mixed Functional Encryption . . . 19

4 Attribute-Based Slotted Inner-Product Functional Encryption **20**
- 4.1 Definition . . . 20
- 4.2 Construction of AB-sIPFE for Arithmetic Branching Programs . . . 21
  - 4.2.1 Arithmetic Key Garbling for ABPs . . . 23
  - 4.2.2 The One-Key Core . . . 24
  - 4.2.3 Bootstrapping the One-Key Core to Public-Key AB-sIPFE . . . 26
  - 4.2.4 Security Analysis . . . 28
- 4.3 Construction of Predicate-Hiding Mixed AB-sIPFE . . . 32

5 Decentralized Multi-Authority Two-Mode Attribute-Based Encryption **33**
- 5.1 Definition . . . 33
- 5.2 Construction with Indistinguishability-Based Security . . . 35
  - 5.2.1 Construction . . . 35
  - 5.2.2 Security Analysis . . . 38
- 5.3 Transformation to Decoder-Based Security . . . 43
- 5.4 Special Case: Multi-Authority Embedded-Identity Private Linear Broadcast Encryption . . . 48

6 Decentralized Multi-Authority Attribute-Based Traitor Tracing **49**
- 6.1 Definition . . . 49
- 6.2 Construction of Secret-Key MA-AB-TT with Embedded Identities . . . 52
  - 6.2.1 Construction of Risky and Limited MA-AB-TT . . . 52
  - 6.2.2 Eliminating Riskiness . . . 56
  - 6.2.3 Eliminating Limitedness . . . 60
  - 6.2.4 Putting It All Together . . . 64
- 6.3 Construction of Public-Key MA-AB-TT . . . 66

A Construction of Predicate-Hiding AB-sIPFE for Comparisons with O(N)-Size Ciphertexts **75**

1 Introduction

Decentralized cryptography. Decentralized cryptography is a broad paradigm for distributing cryptographic trust across multiple independent entities, each contributing a share of the information, authority, or computational capability needed to realize a cryptographic task. At the level of basic primitives, this paradigm is exemplified by threshold cryptography, where secret material is distributed among several parties and operations such as key generation, signing, or public-key decryption can be carried out only through an authorized collaboration, yielding threshold signatures [GJKR96b, GJKR96a, CGJ+99, Bol03, KG20] and threshold encryption/decryption [DF90, SG98, CG99, FP01, BBH06]. The practical importance of this direction is further reflected in NIST's recent efforts toward standardizing multi-party threshold schemes [BMV19, BDV20, BP26]. At a more advanced policy-driven level, prominent examples of decentralization include multi-authority attribute-based encryption [Cha07, MKE09, LCLS08, CC09, LW11, RW15, WWW22, DKW23, AG23, DTV25], more general multi-authority functional encryption [GY24, GY25], and distributed anonymous credential systems [LRSW99, CL01].

Traitor tracing. In this paper, we focus on the setting of traitor tracing (TT) [CFN94, BSW06, BZ14, GKW18, CVW+18, GKW19, Zha20, GLW23, BPR24, Zha25, BHO25, BBF+25, DDP+25], a cryptographic primitive for protecting digital content against unauthorized redistribution. In a traitor tracing scheme, users are issued distinct secret decryption keys that enable access to protected content, while a tracing algorithm allows a distributor to identify one or more malicious users responsible for constructing or distributing a pirate decoder. A fundamental limitation of the traditional notion of traitor tracing is its reliance on a single trusted authority responsible for generating and distributing user secret keys, as well as performing tracing operations. This centralized trust model raises significant security and privacy concerns: a compromised or malicious authority may falsely implicate innocent users during the tracing process, and the ability of a single entity to trace users arbitrarily introduces inherent risks of surveillance and abuse.

Extension to multiple authorities. To address these limitations, we initiate the study of multi-authority traitor tracing (MA-TT), a natural extension of traitor tracing to a distributed trust setting. In this model, tracing capabilities are no longer concentrated in a single entity; instead, multiple independent tracing authorities each manage their own user populations and corresponding key material. A ciphertext is associated with a policy over these authorities, specifying which subsets of authorities must participate in a tracing procedure in order to implicate a user. Decryption requires a valid combination of secret keys—issued by different tracing authorities— which satisfies the policy of the ciphertext. Tracing requires a structured collaboration among the tracing authorities: only when an authorized coalition of tracing authorities jointly corroborates evidence can a user be identified as a contributor to a pirate decoder. This enables fine-grained control over who can trace whom, while preventing any single authority from unilaterally performing surveillance or framing users.

Below, we provide several motivating examples illustrating the benefits of allowing encryption under conjunction, disjunction as well as more general (monotone) policies over multiple tracing authorities.

- **Example 1: Conjunction policy.** Consider a conjunction policy in a corporate insider investigation setting, where sensitive engineering documents are leaked. Employees receive traceable decryption keys, but tracing power is distributed across independent parties such as HR, corporate security, and legal or compliance teams. Since each of these entities may have conflicting incentives, no single authority is trusted on its own. A user can only be accused if all tracing authorities independently derive the same result, ensuring that no single compromised or adversarial office can fabricate evidence or retaliate against individuals.
- **Example 2: Disjunction policy.** In contrast, under a disjunction policy, consider a cross-border media distribution consortium jointly operated by a European provider, a US provider, and an Asia-Pacific telecom group. Each authority manages only its own subscribers due to privacy regulations and jurisdictional constraints, and no single entity is allowed to trace users globally. In this setting, tracing is naturally partitioned: each authority can only test whether a suspected pirate decoder involves one of its own users, and tracing can proceed if any one of the authorized regional authorities identifies a match. Consequently, European authorities learn nothing about US or Asia-Pacific subscribers, while each region retains the ability to trace pirate decoders involving its own users.
- **Example 3: General monotone policy.** Extending the previous example, suppose that different jurisdictions impose different regulatory requirements on tracing. In particular, assume that EU regulations

require that any tracing involving European users must be approved by an independent external auditor, while US and Asia-Pacific providers are allowed to perform tracing under their own internal compliance procedures. As a result, tracing is no longer governed by a uniform structure across authorities. A natural policy in this setting is

$$(\mathsf{EU} \wedge \mathsf{Auditor}) \vee \mathsf{US} \vee \mathsf{APAC}.$$

Here, tracing European users requires the joint participation of the EU provider and an external auditor, whereas tracing US or Asia-Pacific users can be carried out by the corresponding regional provider alone. This asymmetric structure captures the fact that different jurisdictions may impose different levels of oversight, and motivates tracing mechanisms that can express arbitrary monotone combinations of regional authorities and auditors within a single system.

- **Example 4: Threshold policy.** As a further example, consider a global content protection system operated by a consortium of large streaming platforms. Each platform issues traceable decryption keys to its subscribers, but no single provider is fully trusted to perform tracing due to potential conflicts of interest and competitive concerns. At the same time, requiring full agreement among all providers may be too restrictive in practice, especially when some parties are unavailable or unwilling to participate in a tracing procedure. To balance these requirements, tracing is governed by a threshold policy, where any coalition of at least k out of n tracing authorities is sufficient to accuse a user of constructing a pirate decoder. In this setting, different subsets of providers can independently initiate tracing, and as long as enough independent authorities observe consistent evidence, a user can be identified as a traitor.

On an abstract level, these examples suggest the following general comparison between conjunction and disjunction constraints. Conjunction policies, as illustrated in Example 1, require agreement among multiple tracing authorities in order to accuse a user. This provides strong protection against unilateral accusations, since no single authority is sufficient to finalize a tracing decision. At the same time, conjunction policies give each individual authority broad visibility: every authority can, in principle, run the tracing procedure with respect to all users, but any resulting accusation only becomes valid if it is corroborated by other authorities. In contrast, disjunction policies, as in Example 2, restrict each authority to its own user population, thereby limiting the scope of tracing information available to any single party. However, within its own jurisdiction, each authority can independently identify and accuse users without requiring agreement from others. More generally, threshold policies and other monotone access structures interpolate between these two extremes by combining conjunction and disjunction constraints. This allows a more flexible trade-off between how widely tracing information is available and how much agreement is required to accuse a user.

Defining multi-authority traitor tracing. We now give a high-level intuition of our definition. In an MA-TT system, a collection of tracing authorities issues user secret keys that are required for decryption and enable traceability of the decryption process. Ciphertexts are associated with a tracing policy over the authorities, specifying which subsets are authorized to jointly accuse a user of constructing a pirate decoder. Decryption is possible only with a collection of keys whose issuing authorities satisfy the policy associated with the ciphertext.

Each tracing authority tid can independently execute a tracing algorithm with black-box access to a decoder box D in order to identify a set of users J_{tid} that contributed to the construction of D. Let $\mathbb{T}$ be a monotone access structure over a set T of tracing authorities, and suppose that D can distinguish encryptions of two messages (μ_0, μ_1) under $\mathbb{T}$ with high probability. Then our notion provides the following guarantees:

(i) Correct tracing with honest authorities: Consider sets $T' \subseteq T$ that are authorized with respect to $\mathbb{T}$ such that for each $\mathsf{tid} \in T'$, the adversary either obtains a secret decryption key or tid is corrupted. If each such set T' contains an authorized subset $T'_{\mathrm{hon}} \subseteq T'$ of honest authorities, then there exists a nonempty set of users J' such that for every $\mathsf{tid} \in T'_{\mathrm{hon}}$, we have $J' \subseteq J_{\mathsf{tid}}$. That is, every authority in T'_{hon} identifies all users in J' as contributors to D.

(ii) Sound tracing: Honest tracing authorities never accuse innocent users.

Tracing outcomes J_{tid} produced by individual authorities are combined in a canonical way. Specifically, one identifies an authorized subset $T' \subseteq T$ such that

$$J' \coloneqq \bigcap_{\mathsf{tid} \in T'} J_{\mathsf{tid}} \neq \emptyset,$$

and declares all users in J' as traitors. If multiple authorized subsets $T'_1, T'_2, \ldots \subseteq T$ yield nonempty intersections $J'_1, J'_2, \ldots$, then all users in the union $\bigcup_i J'_i$ can be identified as traitors. Guarantee (i) ensures that this procedure identifies at least one valid traitor whenever decryption requires keys from an authorized set of honest authorities, while guarantee (ii) ensures that no innocent user is ever falsely accused unless the adversary corrupts a set of tracing authorities that itself is authorized.

In addition to the above setting with secret traceability, where the tracing algorithm requires the secret key of an authority (and can therefore only be executed by that authority), we also consider a public traceability variant. In this setting, the tracing algorithm only takes the public keys of the tracing authorities as input and can therefore be executed by any party. Finally, we note that both our secret-key and public-key notions of multi-authority traceability naturally recover the standard traceability definition of centralized TT schemes when specialized to the case of a single honest authority and the trivial tracing policy, i.e., $T = \{\mathsf{tid}\}$ and $\mathbb{T} = \{T\}$.

Attribute-based traitor tracing. Traditional traitor tracing is as an extension of public key encryption, where users receive distinct secret keys that are additionally endowed with traceability. As a generalization, existing works [LCW13,LW15,NCD+16,CVW+18] consider attribute-based traitor tracing (AB-TT), in which the underlying encryption primitive is attribute-based encryption (ABE), viewed as a generalization of public-key encryption. In more detail, in a (ciphertext-policy) AB-TT scheme users receive attribute-based secret keys that allow them to decrypt ciphertexts whose policies are satisfied by their attributes, while a tracing procedure enables an authority to identify users whose keys were used to construct a pirate decoder. All existing AB-TT schemes are inherently centralized, relying on a single trusted authority for both key generation and tracing.

To overcome this limitation, we consider multi-authority traitor tracing in the attribute-based setting. Even more general, we introduce the notion of multi-authority AB-TT (MA-AB-TT), which combines the distributed tracing functionality of MA-TT with the decentralized access control of multi-authority ABE (MA-ABE). Recall that in an MA-ABE scheme, secret key issuance is distributed across multiple authorities, each responsible for its own set of attributes. Encryption is performed with respect to an access policy over attributes that may span several authorities, while decryption is possible only for users whose combined attributes satisfy the policy. Our MA-AB-TT primitive combines multi-authority access control and multi-authority traceability in a single system. Specifically, ciphertexts are equipped with *two* independent policies. The first is an attribute policy, defined over a set of attributes that may be controlled by multiple attribute authorities, as in MA-ABE. The second is a tracing policy, defined over a separate collection of tracing authorities, specifying which subsets of tracing authorities are authorized to jointly accuse a user of contributing to a pirate decoder, as in the MA-TT notion introduced above. Following the decentralized MA-ABE formulation of [LW11], the setup phase of MA-AB-TT does not require any interaction among authorities, but authorities can come into existence independently at any time by publishing their own public key.

1.1 Our Results

We initiate the study of traitor tracing in a fully decentralized setting, introducing both multi-authority traitor tracing (MA-TT) and its attribute-based generalization (MA-AB-TT).

We construct schemes supporting arbitrary monotone access structures with adaptive security against static corruption of authorities. Security is proven under the standard matrix decisional Diffie-Hellman (MDDH) assumption in the random oracle model.[1] In the setting where tracing is performed by authorities, our construction meets the best known asymptotic performances of centralized TT schemes. In particular, all system parameters grow only logarithmically in the number of users. In the case of public traceability, we provide a feasibility result where authority public keys and ciphertexts grow linearly with the number of users.

Theorem 1.1 (Informal). *Assuming the hardness of MDDH on pairings, there exist MA-AB-TT schemes in the random oracle model that are secure against adaptive key queries and adaptive challenge ciphertexts under static corruption of authorities. The schemes support arbitrary monotone access structures for both access control and tracing. Moreover, they achieve one of the following tradeoffs:*

- ***Secret traceability:*** *Authority public keys and user secret keys have size $O(1)$, while ciphertexts have size $O(|\mathbb{A}| + |\mathbb{T}|)$.*

[1] Essentially all known MA-ABE schemes are proven in the random oracle model. Since our MA-(AB-)TT construction in particular implies an MA-ABE for all monotone policies, it falls into the same regime. The only exception is [WWW22], which achieves standard-model security under evasive LWE but supports only subset policies.

Work	Traceability	Policy	Assumption	Model	K	N	$\lvert\mathsf{pk}\rvert$	$\lvert\mathsf{sk}\rvert$	$\lvert\mathsf{ct}\rvert$	τ_{Enc}
[GSY26, Section 5.3]	private	$\bigwedge_K$ [†]	LWE + pairings	ROM	B	B	1	1	K	$\mathsf{poly}(K, N)$
(ours) Section 6.2	private	LSSS	MDDH (on pairings)	ROM	U	U	1	1	$\lvert\mathbb{T}\rvert$	$\lvert\mathbb{T}\rvert$
[GSY26, Section 7.3]	public	$\bigwedge_K$	MA-FE for P/Poly	standard	B	B [‡]	1	1	1	1
[GSY26, Section 7.4]	public	$\bigwedge_K$	PKE	standard	B	B	N	1	$K \cdot N$	$K \cdot N$
(ours) Section 6.3	public	LSSS	MDDH (on pairings)	ROM	U	B	N	1	$\lvert\mathbb{T}\rvert \cdot N$	$\lvert\mathbb{T}\rvert \cdot N$

[†] $\bigwedge_K$ denotes conjunction over exactly K authorities.
[‡] The construction likely extends to the unbounded setting under a suitable generalization of the model of [GSY26].

Table 1: Comparison of our constructions (viewed as MA-TT schemes) with concurrent work [GSY26]. Here, K and N are the numbers of authorities and users in the system, respectively. B and U indicate if K and N are polynomially bounded during setup or unbounded. $\lvert\mathsf{pk}\rvert$, $\lvert\mathsf{sk}\rvert$, $\lvert\mathsf{ct}\rvert$ and τ_{Enc} refer to the sizes of authority public keys, user secret keys, ciphertexts and the encryption time. $\lvert\mathbb{T}\rvert$ denotes the size of the tracing policy $\mathbb{T}$ with respect to which encryption is performed. In particular, if $\mathbb{T} = \bigwedge_K$, then $\lvert\mathbb{T}\rvert = K$. We hide factors in $\mathsf{poly}(\lambda, \log K, \log N)$.

- ***Public traceability:*** *Authority public keys have size $O(N)$, user secret keys have size $O(1)$, and ciphertexts have size $O((\lvert\mathbb{A}\rvert + \lvert\mathbb{T}\rvert) \cdot N)$.*

Here, K and N denote the maximum number of authorities and users in the system, while $\lvert\mathbb{A}\rvert$ and $\lvert\mathbb{T}\rvert$ refer to the sizes of the access-control and tracing policies under which a ciphertext is generated. The $O(\cdot)$ notation hides factors polynomial in λ, $\log K$, and $\log N$.

We compare our results with the concurrent work [GSY26] in Table 1 and Section 1.2 below.

Our approach. Our construction builds on the blueprint for optimal centralized traitor tracing due to [GKW18, CVW+18, Zha25]. At a high level, these works combine ABE with mixed functional encryption (MFE) to obtain private linear broadcast encryption (PLBE), from which traitor tracing follows generically. Our main technical contribution is to lift this paradigm to the multi-authority setting. To this end, we introduce a multi-authority analogue of PLBE. Our construction is inspired by the framework of [AG23], which realizes MA-ABE by letting each authority independently instantiate an inner-product functional encryption (IPFE) scheme.[2] We adopt this modular approach in the tracing setting by augmenting IPFE with MFE functionality, yielding a primitive we call mixed attribute-based IPFE (mixed AB-IPFE). Each authority then runs their own mixed AB-IPFE instance and can locally execute the tracing algorithm.

To realize mixed AB-IPFE, we follow the general paradigm of combining ABE with MFE, but replace the ABE component with attribute-based IPFE. While several such constructions are known [ACGU20, AGT21, LLW21, PD21, Tom23, DP23, ATY23], our application requires the more complex slotted IPFE (sIPFE) functionality [LL20a], which combines features of standard public-key IPFE and secret-key function-hiding IPFE. To the best of our knowledge, the only existing construction capturing this IPFE variant is due to [Tom23] which, however, supports only inner-product predicates as access policies. Since this policy class is insufficient for our application, we provide a new attribute-based sIPFE scheme supporting arithmetic branching programs. This construction might be of independent interest.

As an additional consequence, our techniques yield a new construction of MA-ABE from sIPFE. This improves upon [AG23], which relies on simulation-secure IPFE and achieves either adaptive security in the generic group model (GGM) or selective security under the SXDH assumption. In contrast, we rely only on indistinguishability-based security of sIPFE and achieve adaptive security without resorting to the GGM, resolving an open problem of [AG23]. To the best of our knowledge, adaptively secure MA-ABE from standard assumptions was previously only known via direct group-based constructions.

Implications to AB-TT. Centralized AB-TT has been studied in both the ciphertext-policy and key-policy setting. In the ciphertext-policy setting, the pairing-based constructions of [LCW13, LW15, NCD+16] achieve traceability only against "strong" decoder boxes, namely pirate decoders that can decrypt encryptions of random messages with high probability. In contrast, the lattice-based construction of [CVW+18] achieves a stronger notion of traceability in the key-policy setting, where it suffices for a decoder box to distinguish encryptions of

[2] In an IPFE scheme, secret keys and ciphertexts are generated with respect to vectors, and decryption reveals their inner product while hiding all other information about the ciphertext vector [ABDP15, ALS16, Wee17, LV16, Lin17, LL20a].

two chosen messages, together with asymptotically optimal parameters. However, that construction does not directly extend to the ciphertext-policy setting, except via generic transformations based on universal circuits.

In contrast, when instantiated with a single tracing authority, our MA-AB-TT construction immediately yields a ciphertext-policy AB-TT scheme achieving the stronger indistinguishability-based notion of traceability and asymptotically optimal parameters. Moreover, it supports exponentially large user spaces via embedded-identity traitor tracing techniques [GKW19], whereas all prior AB-TT constructions are restricted to index-based traceability over polynomial-size user sets.

1.2 Concurrent and Related Works

Existing models for decentralized traitor tracing. To the best of our knowledge, only three works [Luo24, BLM+24, CMMS25] have studied traitor tracing without relying on a central authority, under the notions of *ad-hoc* and *registered* traitor tracing. In these models, each user generates their own key pair and publishes the corresponding public key, so that the secret key is known only to the user, thereby naturally eliminating the key escrow problem.

Despite this appealing property, these approaches come with several practical and efficiency drawbacks. First, users are required to publish their public keys on a public bulletin board, and a (untrusted) curator must then aggregate all public keys into a single succinct master public key. Building and maintaining such an infrastructure introduces non-trivial deployment complexity. Second, encryption requires access to the most recent master public key, meaning that encryptors must continually update their view of the system. In particular, ciphertexts cannot be prepared in advance for users who join the system at a later time. Third, known constructions inherit the same asymptotic inefficiencies as centralized traitor tracing schemes with public traceability (see below).

Importantly, the key escrow problem and the risk of false accusation are *not* resolved by traditional TT schemes with public traceability [CPP05, BW06, PST06, BZ14]. Although tracing in such schemes can be performed using only public information and without relying on a tracing authority, they still require a trusted party for setup and key generation. Similar issues arise in a recent line of work on threshold TT schemes [BPR24, BHO25, CPP25, BBF+25, DDP+25, BFO26]. While these schemes distribute the decryption procedure across multiple parties, tracing is still performed either by a tracing authority or publicly, and thus inherits the concerns of centralized TT schemes.

Parameter sizes in traitor tracing. Since the introduction of traitor tracing by Chor, Fiat, and Naor [CFN94], a central goal has been to minimize the sizes of public keys, secret keys, and ciphertexts as a function of the number of users n. The first sublinear scheme with secret traceability, due to Boneh, Sahai, and Waters [BSW06], achieves parameter sizes of $O(\sqrt{n})$ using pairings. Subsequent pairing-based constructions reduced this to $O(n^{1/3})$ [Zha20, GLW23], and very recently Zhandry [Zha25] obtained optimal parameters of size $\mathrm{poly}(\lambda, \log n)$ from pairings. Optimal parameters were previously known only from indistinguishability obfuscation and functional encryption [GGH+13, BZ14] as well as from LWE [GKW18, CVW+18]. We also note that some schemes optimize only ciphertext size, achieving constant-size ciphertexts at the cost of secret keys that grow with n [BN08, BP08].

In the public traceability setting, the most efficient schemes with asymptotically optimal parameters rely on indistinguishability obfuscation or functional encryption [GGH+13, BZ14, GVW19], whereas pairing-based public-tracing schemes [BW06, CPP05, PST06] achieve only suboptimal parameters that grow superlogarithmically in n. Obtaining optimal public traceability without obfuscation remains an interesting open problem, even in the presence of a central authority. A similar dichotomy between public and secret tracing, with various parameter trade-offs, also appears in the aforementioned works on threshold traitor tracing [BPR24, BHO25, CPP25, BBF+25, DDP+25, BFO26]; we refer to [DDP+25] for a detailed comparison.

Comparison with the concurrent work [GSY26]. A concurrent and independent work of Goyal, Snyder, and Yadugiri [GSY26] also considers traitor tracing in the multi-authority setting. Below we outline how our definitions and constructions improve upon those of [GSY26]. In fact, our work addresses several of the open problems listed in [GSY26].

- **Definition:** Both works formalize multi-authority traitor tracing, but differ in modeling choices. We highlight two differences. First, the tracing algorithm in [GSY26] simultaneously takes as input the master secret keys of all authorities. In contrast, our tracing algorithm only requires the master secret key of

a *single* authority at a time together with the public keys of the remaining authorities. This enables a non-interactive tracing procedure in which partial outputs can be computed independently and aggregated in a publicly observable way. This realizes a true decentralization of the tracing procedure which is absent in [GSY26]. Second, the traceability notion of [GSY26] corresponds, in our terminology, to a conjunction policy over a fixed number K of authorities; our definition extends this to arbitrary monotone policies. In fact, we recover the definition of [GSY26] by specializing our traceability notion to the case of a fixed policy—the conjunction over K authorities—and viewing the local executions of the tracing algorithm by all authorities as a single execution of a joint tracing procedure. Moreover, [GSY26] does not consider the extension to the attribute-based setting.

- **Constructions:** We refer to Table 1 for an apples-to-apples comparison of the constructions in [GSY26] to those presented in this work. We now show how our (secretly traceable) construction addresses the five open problems listed in [GSY26, page 12] in order.

  1. We partially resolve the quest for succinct ciphertexts. The size of our ciphertexts is independent of the number of authorities K and users N in the system; however, it still depends on the size of the tracing policy relative to which encryption is performed. This dependency is an artifact of the underlying AB-sIPFE for ABPs used to instantiate the MA-TT scheme.[3]
  2. We construct MA-TT using only the MDDH assumption on pairings, without additionally relying on lattice-based assumptions. Moreover, encryption and decryption complexity are independent of K and N and only depend on the size of the tracing policy.
  3. We show that techniques from embedded-identity traitor tracing [GKW19] are applicable in the multi-authority setting, yielding support for an unbounded user space without pre-registration.
  4. Our construction extends the class of supported access structures from conjunctive policies (the only setting captured by [GSY26]) to arbitrary monotone access structures; in particular, this resolves the open problem of supporting threshold (quorum) policies.
  5. We partially address the problem of traceability in the presence of corrupted authorities. Our notion guarantees successful tracing provided that among the secret keys used in the decoder there exists an authorized subset whose issuing authorities are all honest. For conjunctive policies this reduces to requiring honesty of all authorities, matching the behavior of [GSY26]. For more relaxed policies, however, tracing still succeeds even if some authorities are corrupted, as long as they do not intersect the relevant authorized set.

2 Technical Overview

For simplicity, we restrict this high-level overview to the case of secretly traceable MA-TT. The extensions to the attribute-based setting and public traceability are conceptually natural, and the full treatment is deferred to the formal sections. Moreover, we describe the construction only for a polynomial-size user space here, whereas the full scheme supports exponentially many users.

We present our construction in several steps. In Section 2.1, we show how to generically build MA-TT (with some limitations) from a new primitive that we call multi-authority private linear broadcast encryption (MA-PLBE). In Section 2.2, we present a generic construction of MA-PLBE from mixed functional encryption (MFE) and attribute-based slotted inner-product functional encryption (AB-sIPFE). The latter is also constructed in this work from plain sIPFE and arithmetic key garbling schemes (AKGS); see Section 2.3 for a discussion. In Section 2.4, we discuss how to generically remove all limitations of the MA-TT scheme from Section 2.1. The full chain of primitives and compilers leading to our MA-TT scheme with polynomial-size user space is summarized in Figure 1.

2.1 A Generic Compiler for MA-TT

We first review an existing compiler for centralized TT which is a special case of MA-TT with a single authority.

[3] The AB-sIPFE scheme in this work (Construction 4.7) is based on [LL20a]. We believe that succinctness can be improved by incorporating techniques from follow up works, e.g. [LL20b, LLL22], and we leave it as an interesting open problem.

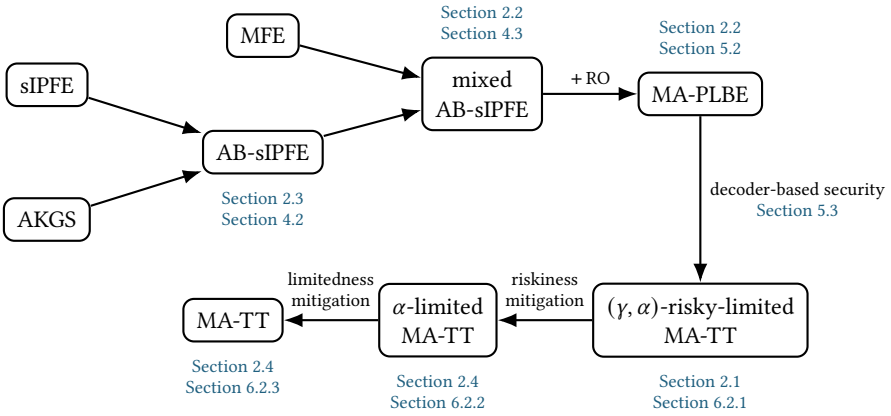

Figure 1: Roadmap of our multi-authority traitor tracing (MA-TT) construction with secret tracing

Recap: Centralized TT. A classic transformation in the literature [BSW06] shows how to generically derive traitor tracing from PLBE. A PLBE scheme for a polynomial number n of users $\mathsf{gid}_1, \ldots, \mathsf{gid}_n$ consists of a public key pk, a master secret key msk, and user secret keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_n$. The system supports two encryption modes. In the *public-key mode*, the encryption algorithm EncPK takes only the public key pk and a message μ, and produces a ciphertext decryptable by every user key sk_i. In the *secret-key mode*, the encryption algorithm EncSK additionally takes the master secret key msk and an index $i \in [0, n]$. A ciphertext encrypted to index i can then only be decrypted by secret keys sk_j with $j > i$. PLBE security is captured by three indistinguishability properties. First, *index hiding* requires that an adversary possessing all secret keys except sk_i cannot distinguish ciphertexts encrypted to indices $i - 1$ and i. Second, *message hiding* states that encryptions of two messages μ_0, μ_1 to index n are computationally indistinguishable. Finally, *mode indistinguishability* requires that a public-key encryption of μ is indistinguishable from a secret-key encryption of μ to index 0.

The traitor tracing construction is immediate: the setup algorithm generates a PLBE instance and distributes sk_j to user gid_j. To trace a pirate decoder D, the tracing algorithm estimates the probability that D correctly decrypts ciphertexts encrypted to index i, for every $i \in [0, n]$. By mode indistinguishability, if D decrypts public-key ciphertexts with probability ε, then it must also decrypt ciphertexts to index 0 with approximately the same probability. On the other hand, message hiding implies that ciphertexts encrypted to index n are undecryptable. Consequently, there exists some index $i \in [n]$ at which the success probability of D drops noticeably when moving from encryptions to index $i - 1$ to encryptions to index i. By index hiding property of PLBE, such a gap can only occur if D internally uses the key sk_i. The tracing algorithm therefore identifies user gid_i as a colluder.

Our MA-TT compiler. To adapt the PLBE-based approach to the multi-authority setting, we introduce a multi-authority extension of PLBE (MA-PLBE)[4] and show that the standard PLBE-to-TT compiler lifts to this decentralized setting. At a high level, MA-TT can be viewed as an MA-ABE system augmented with traceability. Indeed, in MA-TT an encryptor produces a ciphertext for a message μ under a tracing policy $\mathbb{T}$ over a set T of tracing authorities. A user gid holding secret keys $\{\mathsf{sk}_{\mathsf{tid},\mathsf{gid}}\}_{\mathsf{tid} \in T'}$ can decrypt whenever $T' \in \mathbb{T}$. Ignoring traceability and viewing the tracing policy as an attribute-based access control policy, this is simply a multi-authority ABE scheme.

Towards enabling tracing, we augment this MA-ABE functionality with a PLBE-style secret-key encryption algorithm. Specifically, we require that each tracing authority tid can, using its master secret key (and the public keys of the remaining authorities), generate ciphertexts in a secret-key mode that additionally depend on a user index $i \in [0, n]$. These ciphertexts are constructed so that a user gid_j can decrypt using keys from T' if and only if $\mathsf{tid} \notin T'$ and $j > i$. We call the resulting primitive multi-authority PLBE.

Given MA-PLBE, the construction of MA-TT follows by a direct lifting of the PLBE-to-TT compiler: setup, key generation, encryption, and decryption are identical to MA-PLBE, and tracing is performed by evaluating a decoder on secret-key encryptions across all indices $i \in [0, n]$. More precisely, an authority tid estimates the success probability of a decoder D on ciphertexts encrypted to index i. If D was built using a secret key issued by tid, then—as in the single-authority setting—there exists an index at which a noticeable drop in success

[4] MA-PLBE is a special case of mixed MA-2ABE introduced in Section 5.

probability occurs, revealing the corresponding key index. If instead D was constructed only from keys issued by a set T' with $\mathsf{tid} \notin T'$, then such a gap need not be detectable by tid. This is consistent with the MA-TT notion, which only requires that there exists at least one authorized subset of tracing authorities that can successfully identify a traitor, rather than requiring every individual authority to be able to trace.

Looking ahead, we do not know how to construct an MA-PLBE scheme with a sufficiently strong notion of mode indistinguishability to directly obtain an MA-TT scheme satisfying full traceability. The problem is that in our construction, mode indistinguishability—namely, indistinguishability between public-key encryptions and secret-key encryptions to index 0—only holds in the absence of additional secret-key ciphertexts for the adversary. However, a straightforward MA-TT compiler à la [GKW18] would require at least one such additional secret-key ciphertext. As a solution, we show that MA-PLBE schemes with this weaker form of mode indistinguishability can still be lifted to MA-TT with a restricted notion of traceability. We then show how to upgrade this construction to full traceability. Similar obstacles already arise in previous pairing-based constructions of (threshold) TT [Zha25, DDP+25], and overcoming them in the multi-authority setting requires a careful generalization of techniques developed in those works. We defer details to Section 2.4.

2.2 Constructing Multi-Authority PLBE

In this section, we present our construction of MA-PLBE. Building on the observation that MA-PLBE can be viewed as an MA-ABE scheme augmented with a secret-key encryption capability, we first revisit an existing MA-ABE scheme due to [AG23]. We then show how this scheme can be naturally extended to support a compatible secret-key encryption algorithm, thereby yielding the desired MA-PLBE instantiation. While other MA-ABE schemes from the literature would likely admit similar extensions, the construction of [AG23] is particularly appealing to work with, thanks to its modular structure and reliance on the IPFE machinery.

2.2.1 A Modular MA-ABE Scheme from Slotted IPFE

Let $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathrm{t}, g_1, g_2, g_\mathrm{t}, e, p)$ be a pairing group. For $i \in \{1, 2, \mathrm{t}\}$ and $a \in \mathbb{Z}_p$, we write $[\![a]\!]_i = g_i^a$ and use additive notation for the group law. The construction relies on inner-product functional encryption (IPFE). An IPFE ciphertext encrypting a vector $[\![\mathbf{u}]\!]_1 \in \mathbb{G}_1^n$ is denoted by $\mathsf{ict}([\![\mathbf{u}]\!]_1)$, while a secret key for a vector $[\![\mathbf{v}]\!]_2 \in \mathbb{G}_2^n$ is written as $\mathsf{isk}([\![\mathbf{v}]\!]_2)$. Decryption reveals only the inner product $[\![\mathbf{v}^\top \mathbf{u}]\!]_\mathrm{t}$ encoded in the target group. When several IPFE instances are used in parallel, we index them accordingly, e.g., $\mathsf{ict}_j(\cdot)$ and $\mathsf{isk}_j(\cdot)$.

An overview of [AG23]. The construction of [AG23] builds on a standard paradigm for *single-authority* ABE [AWY20, AG21, LLL22, HLL24] that combines pairing-based IPFE with linear secret sharing schemes (LSSS).

Consider an access structure $\mathbb{A}$ over a set of attributes A represented by an LSSS matrix $(\mathbf{M} \in \mathbb{Z}_p^{n \times \ell}, \rho)$, where $\rho\colon [\ell] \to A$ labels columns by attributes. To encrypt a message $[\![\mu]\!]_\mathrm{t} \in \mathbb{G}_\mathrm{t}$ under $\mathbb{A}$, one generates two secret sharings

$$(\sigma_1, \ldots, \sigma_\ell) \coloneqq (\sigma, \mathbf{s}) \cdot \mathbf{M} \qquad \text{and} \qquad (\tau_1, \ldots, \tau_\ell) \coloneqq (0, \mathbf{t}) \cdot \mathbf{M},$$

where $\sigma \leftarrow\!\!\$\ \mathbb{Z}_p$ and $\mathbf{s}, \mathbf{t} \leftarrow\!\!\$\ \mathbb{Z}_p^{n-1}$. These shares are then encrypted using independent IPFE instances. To generate a key for a set of attributes $S \subseteq A$, one samples $r \leftarrow\!\!\$\ \mathbb{Z}_p$ and generates IPFE secret keys for those instances corresponding to S:

$$\left.\begin{array}{ll} \mathsf{cp\text{-}abe.sk}_S: & \{\mathsf{isk}_j([\![(1, r)]\!]_2)\}_{j \in J} \\ \mathsf{cp\text{-}abe.ct}_\mathbb{A}: & \{\mathsf{ict}_j([\![(\sigma_j, \tau_j)]\!]_1)\}_{j \in [\ell]},\ [\![\sigma + \mu]\!]_\mathrm{t} \end{array}\right\} \implies \{[\![\sigma_j + r \cdot \tau_j]\!]_\mathrm{t}\}_{j \in J},\ [\![\sigma + \mu]\!]_\mathrm{t},$$

where $J = \{j_1, \ldots, j_{|J|}\} \coloneqq \rho^{-1}(S)$. IPFE decryption yields encodings of $\sigma_j + r \cdot \tau_j$ for all $j \in J$. If $S \in \mathbb{A}$, there exists a coefficient vector $\mathbf{c} = (c_1, \ldots, c_{|J|})$ such that $\mathbf{M}_J \cdot \mathbf{c}^\top = \mathbf{e}_1^\top$, where $\mathbf{M}_J$ denotes the restriction of $\mathbf{M}$ to columns indexed by J. Then the masking term $[\![\sigma]\!]_\mathrm{t}$ can be recovered as

$$\sum_{i \in |J|} c_i \cdot [\![\sigma_{j_i} + r \cdot \tau_{j_i}]\!]_\mathrm{t} = [\![\sigma]\!]_\mathrm{t}.$$

Note that using a second sharing $\{\tau_j\}_j$ is essential. Indeed, directly pairing the shares σ_j with the randomness r in the keys would only yield $[\![r \cdot \sigma]\!]_\mathrm{t}$, whereas the goal is to recover $[\![\sigma]\!]_\mathrm{t}$ itself.

To turn the above into an MA-ABE scheme, we decentralize key generation across independent authorities. Each attribute authority $\mathsf{aid} \in A$[5] runs its own IPFE instance to issue keys. Crucially, ciphertext generation remains unchanged; only key generation is distributed. To correlate keys issued to the same user across different authorities, we associate each user gid with a shared randomness $r_{\mathsf{gid}} \in \mathbb{Z}_p$, derived non-interactively via a full-domain hash function $H\colon \{0,1\}^* \to \mathbb{G}_2$ as $[\![r_{\mathsf{gid}}]\!]_2 := H(\mathsf{gid})$. Let gid be a user holding attributes from authorities $S \subseteq A$. The MA-ABE scheme is defined as:

$$\left.\begin{array}{lll} \forall \mathsf{aid} \in S, & \mathsf{ma\text{-}abe.sk}_{\mathsf{aid},\mathsf{gid}}: & \mathsf{isk}_{\mathsf{aid}}([\![(1, r_{\mathsf{gid}})]\!]_2) \\ & \mathsf{ma\text{-}abe.ct}_{\mathbb{A}}: & \{\mathsf{ict}_{\rho(j)}([\![(\sigma_j, \tau_j)]\!]_1)\}_{j\in[\ell]},\ [\![\sigma + \mu]\!]_{\mathrm{t}} \end{array}\right\} \implies \{[\![\sigma_j + r_{\mathsf{gid}} \cdot \tau_j]\!]_{\mathrm{t}}\}_{j\in J},\ [\![\sigma + \mu]\!]_{\mathrm{t}}.$$

Decryption proceeds identically to the centralized setting.

Proving security. The security proof of [AG23] relies on IPFE with simulation security. Slightly oversimplified, assume that such IPFE only reveals inner products of the form $\sigma_j + r_{\mathsf{gid}} \cdot \tau_j$, encoded in $\mathbb{G}_2$. Under the DDH assumption in $\mathbb{G}_2$, the collection

$$\left\{[\![(r_{\mathsf{gid}} \cdot \tau_1, \ldots, r_{\mathsf{gid}} \cdot \tau_\ell)]\!]_2\right\}_{\mathsf{gid}}$$

is indistinguishable from a collection of independent random sharings

$$\left\{[\![(\tau_{\mathsf{gid},1}, \ldots, \tau_{\mathsf{gid},\ell})]\!]_2\right\}_{\mathsf{gid}},$$

where $(\tau_{\mathsf{gid},1}, \ldots, \tau_{\mathsf{gid},\ell}) := (0, \mathbf{t}_{\mathsf{gid}}) \cdot \mathbf{M}$ and $\mathbf{t}_{\mathsf{gid}} \leftarrow\!\!\$\ \mathbb{Z}_p^{n-1}$ is freshly sampled for each user gid. Given this view, IND-CPA security of the MA-ABE scheme follows from the information-theoretic one-time security of the underlying LSSS, together with the admissibility condition, which ensures that the adversary never obtains an authorized set of shares for any user gid.

The reliance on simulation-secure IPFE in this proof strategy, however, imposes undesirable restrictions on the resulting MA-ABE scheme. This is because known instantiations of such strong IPFE notions are either proven adaptively secure in the generic group model [LLL22], or achieve only selective security in the standard model [AG23].[6] To obtain adaptive security without resorting to the generic group model, we replace simulation-secure IPFE with an indistinguishability-based alternative. More precisely, we employ slotted IPFE [LL20a], which combines a standard (public) IPFE component used for actual encryption with a second, secret component satisfying a stronger *function-hiding* property. An IPFE scheme is said to be function-hiding [LV16, Lin17] if it does not only hide message vectors $[\![\mathbf{u}]\!]_1$ embedded in ciphertexts $\mathsf{ict}([\![\mathbf{u}]\!]_1)$ but also hides key vectors $[\![\mathbf{v}]\!]_2$ embedded in secret keys $\mathsf{isk}([\![\mathbf{v}]\!]_2)$. The function-hiding component of sIPFE, which is only accessible during the security proof, is used to program the randomized secret sharings

$$\left\{[\![(r_{\mathsf{gid}} \cdot \tau_1, \ldots, r_{\mathsf{gid}} \cdot \tau_\ell)]\!]_2\right\}_{\mathsf{gid}}$$

into a hidden coordinate of the user keys. Importantly, this programming requires only indistinguishability-based security, yet enables the DDH-based randomization step used in the simulation-based proof strategy above. Finally, by viewing the underlying LSSS as a piecewise-secure linear arithmetic key garbling scheme, we obtain adaptive security via a sequence of hybrid games inspired by the security proof of the single-authority ABE in [LL20a]. We refer to Section 5.2 for full details.

2.2.2 Enabling Secret-Key Encryption

Our next step is to augment the scheme from the previous section with an additional secret-key encryption algorithm to indices $i \in [0, n]$. We begin by recalling a known compiler [GKW18, CVW+18, Zha25] that constructs PLBE from a combination of ABE and mixed functional encryption (MFE).

Recap: PLBE from ABE and MFE. MFE is a specialized encryption scheme with two encryption modes: (i) a public-key mode, where encryption uses only the public key pk and outputs a ciphertext ct, and (ii) a secret-key mode, where encryption takes the master secret key msk and a function f and outputs a function-dependent

[5] We assume that each authority controls only one attribute, and hence we can use the words "authority" and "attribute" interchangeably. This definition naturally generalizes to the situation in which each authority potentially controls several attributes [LW11, RW15].

[6] Although standard-model constructions of adaptively simulation-secure IPFE exist [ALMT20], they do not achieve the group-based functionality required here, i.e. they only support ciphertexts and secret keys for vectors $\mathbf{u}, \mathbf{v} \in \mathbb{Z}_p^n$ rather than $\mathbf{u} \in \mathbb{G}_1^n, \mathbf{v} \in \mathbb{G}_2^n$.

ciphertext ct_f. Secret keys sk_x are associated with messages x. A public-key ciphertext ct decrypts to 1 under any secret key sk_x. In contrast, a secret-key ciphertext ct_f decrypts under sk_x to the value $f(x)$. The security of MFE is defined via two notions. First, *mode indistinguishability* guarantees that public-key ciphertexts are indistinguishable from secret-key ciphertexts for function f, except when the adversary holds a secret key sk_x such that $f(x) \neq 1$. Second, *function indistinguishability* ensures that the adversary learns nothing about f beyond the values $f(x)$ for messages x for which it possesses the corresponding secret keys.

To construct PLBE, [GKW18] combines MFE with (ciphertext-policy) ABE as follows.[7] The ABE scheme is used to encrypt messages, where the policy is instantiated by a circuit $C[\mathsf{ct}]$ that runs MFE decryption of a hardwired ciphertext ct. In the public-key mode, the hardwired ciphertext is an MFE public-key ciphertext ct. In the secret-key mode, encryption to an index i uses an MFE secret-key ciphertext $\mathsf{ct}_{>i}$ that encrypts the comparison function $(> i)(\cdot)$, which outputs 1 if and only if the input is greater than i. Each user j receives an ABE secret key whose attribute is an MFE secret key corresponding to index j. By correctness of MFE, a public-key ciphertexts accepts any such key since the embedded MFE decryption outputs 1 on ct. On the other hand, for a secret-key ciphertext $\mathsf{ct}_{>i}$, evaluation yields 1 only for secret keys corresponding to indices $j > i$, and thus only users with index greater than i can decrypt.

As an important detail, we note that the underlying ABE scheme must be expressive enough to accommodate the MFE decryption circuit within its policy language. We come back to this aspect later.

Putting pieces together. The compiler above can be combined with the modular MA-ABE construction from the previous subsection by replacing the ordinary ABE layer with an attribute-based slotted IPFE layer. In more detail, an AB-sIPFE key is associated with an attribute x and an inner-product vector $\mathbf{v}$, while a ciphertext is associated with a predicate f and an inner-product vector $\mathbf{u}$. If $f(x) = 1$, decryption returns $[\![\mathbf{v}\mathbf{u}^\top]\!]_\mathsf{t}$; otherwise it returns $\perp$. This is precisely the interface needed to make the ABE policy depend on an MFE ciphertext while still preserving the linear payload required by the multi-authority construction.

Concretely, in the public-key mode we generate an MFE public ciphertext ct and use the AB-sIPFE predicate corresponding to the circuit

$$C[\mathsf{ct}](\cdot) = \mathsf{MFE.Dec}(\cdot, \mathsf{ct}).$$

In the secret-key mode we instead generate an MFE secret-key ciphertext $\mathsf{ct}_{>i}$ for the function $(> i)(\cdot)$ and use the analogous predicate $C[\mathsf{ct}_{>i}]$. A user key consists of an AB-sIPFE key whose attribute is an MFE key sk_j for the user's index j. Thus, public-mode ciphertexts are accepted by every user key, whereas secret-mode ciphertexts are accepted exactly by those keys for which the predicate $(> i)(\cdot)$ evaluates to 1.

The resulting object is what we call a *mixed* AB-sIPFE. The MFE layer supplies the public/secret mode switch, and the AB-sIPFE layer enforces the predicate $C[\mathsf{ct}]$ or $C[\mathsf{ct}_{>i}]$ while still outputting the same target-group inner product $[\![\mathbf{v}\mathbf{u}^\top]\!]_\mathsf{t}$ needed by the LSSS reconstruction in the MA-PLBE construction. Therefore, the main remaining task is to construct AB-sIPFE for a policy class rich enough to represent the relevant MFE decryption circuits. We do so for arithmetic branching programs (ABPs) by extending the arithmetic-key-garbling-scheme (AKGS) based methodology of [LL20a] to the ciphertext-policy, slotted IPFE setting.

2.3 Construction of Attribute-Based Slotted IPFE

We now give an overview of our AB-sIPFE construction. The exposition follows the same high-level blueprint as the construction of ABE from arithmetic key garbling schemes and IPFE in [LL20a]: we first isolate a one-key secure core construction, then formulate the key-by-key lift to the many-key setting, and finally use slotted IPFE to make the encryption public-key.

The AB-sIPFE interface. Our AB-sIPFE is considered in the ciphertext-policy setting. A secret key is associated with an attribute vector $\mathbf{x} \in \mathbb{Z}_p^n$ and an inner-product vector $\mathbf{v} \in \mathbb{Z}_p^{\mathfrak{s}}$, while a ciphertext is associated with an ABP predicate f and an inner-product vector $\mathbf{u} \in \mathbb{Z}_p^{\mathfrak{s}}$. The slot set is divided as

$$\mathfrak{s} = \mathfrak{s}_{\mathrm{pub}} \cup \mathfrak{s}_{\mathrm{priv}}, \qquad \mathbf{v} = (\mathbf{v}_{\mathrm{pub}}, \mathbf{v}_{\mathrm{priv}}), \qquad \mathbf{u} = (\mathbf{u}_{\mathrm{pub}}, \mathbf{u}_{\mathrm{priv}}).$$

Here $\mathfrak{s}_{\mathrm{pub}}$ denotes the public slot block and $\mathfrak{s}_{\mathrm{priv}}$ denotes the private slot block. The public encryption algorithm only takes $\mathbf{u}_{\mathrm{pub}}$ and implicitly sets the private coordinates to zero, whereas the secret-key encryption algorithm

[7] In fact, the original construction of [GKW18] uses *key-policy* ABE, but the resulting compiler is almost identical.

can encrypt the full vector $\mathbf{u}$. Correctness requires that whenever $f(\mathbf{x}) = 1$, decryption returns

$$[\![\mathbf{v}\mathbf{u}^\top]\!]_t = [\![\mathbf{v}_{\mathrm{pub}}\mathbf{u}_{\mathrm{pub}}^\top + \mathbf{v}_{\mathrm{priv}}\mathbf{u}_{\mathrm{priv}}^\top]\!]_t.$$

As in slotted IPFE, the public slot of a key vector is not hidden, since anyone can locally encrypt public-slot vectors. The private slot is the part used in the proof to move information between keys and ciphertexts.

Arithmetic key garbling. The access-control layer is implemented using an arithmetic key garbling scheme (AKGS) for ABPs. ABP is expressive enough for capturing the MFE decryption circuit mentioned in Section 2.2. For an ABP $f : \mathbb{Z}_p^n \to \mathbb{Z}_p$ and secrets $\alpha, \beta \in \mathbb{Z}_p$, the garbling algorithm outputs affine label functions

$$L_j(\mathbf{X}) = \mathbf{L}_j[\mathsf{const}] + \sum_{i\in[n]} \mathbf{L}_j[\mathsf{coef}_i]\mathbf{X}[i], \qquad j \in [m],$$

where m is linear in the ABP size. On input the evaluated labels $\ell_j = L_j(\mathbf{x})$, the evaluation algorithm computes

$$\mathsf{Eval}(f, \mathbf{x}, \ell_1, \ldots, \ell_m) = \alpha f(\mathbf{x}) + \beta.$$

The garbling is linear in (α, β) and in its randomness. The proof uses the same piecewise simulation properties as in [LL20a]: the first label can be reversely sampled from the target value $\gamma = \alpha f(\mathbf{x}) + \beta$ and the remaining labels, while each later label can be replaced one at a time by a fresh random value without changing the distribution of the unreplaced labels. This property is what lets the proof simulate an unauthorized key/ciphertext interaction without knowing the key attribute in advance.

One-key AB-sIPFE. Let us first ignore the public-key requirement and consider a one-key one-ciphertext setting. The construction uses two independent IPFE copies. The first copy carries the inner-product payload, and the second copy computes the AKGS labels. The vectors used by the honest algorithms have the following form.

	pad	$\mathfrak{s}_{\mathrm{pub}}$	$\mathfrak{s}_{\mathrm{priv}}$
$\mathbf{V}_{\mathrm{ip}}$	1	$\mathbf{v}_{\mathrm{pub}}$	$\mathbf{v}_{\mathrm{priv}}$
$\mathbf{U}_{\mathrm{ip}}$	μ	$\mathbf{u}_{\mathrm{pub}}$	$\mathbf{u}_{\mathrm{priv}}$

	const	coef_1	$\cdots$	coef_n
$\mathbf{V}_{\mathrm{att}}$	1	$\mathbf{x}[1]$	$\cdots$	$\mathbf{x}[n]$
$\mathbf{U}_j$	$\mathbf{L}_j[\mathsf{const}]$	$\mathbf{L}_j[\mathsf{coef}_1]$	$\cdots$	$\mathbf{L}_j[\mathsf{coef}_n]$

The payload copy decrypts to $[\![\mu + \mathbf{v}\mathbf{u}^\top]\!]_t$, while the label copy returns $[\![L_j(\mathbf{x})]\!]_t$ for all $j \in [m]$. If the predicate is $f(\mathbf{x}) \neq 0$, the ciphertext garbles f with secrets $(\alpha, \beta) = (\mu, 0)$; AKGS evaluation gives $\mu f(\mathbf{x})$, from which the decryptor recovers μ. If the predicate is $f(\mathbf{x}) = 0$, the ciphertext garbles with secrets $(\alpha, \beta) = (\eta, \mu)$ for fresh η, so AKGS evaluation gives μ whenever $f(\mathbf{x}) = 0$. In both cases, an authorized decryptor subtracts the recovered pad $[\![\mu]\!]_t$ from the payload output and obtains $[\![\mathbf{v}\mathbf{u}^\top]\!]_t$.

The reason this core is useful is the same as in the one-key ABE of [LL20a]. IPFE computes the affine labels $L_j(\mathbf{x})$ in the exponent, and AKGS security says that, when the predicate is false, these labels reveal only the value $\alpha f(\mathbf{x}) + \beta$, equaling to 0 in the nonzero predicate mode and $\eta f(\mathbf{x}) + \mu$ in the zero predicate mode, which is independent of the hidden pad μ. For adaptive security, the proof cannot hardwire all labels at once because the challenge point $\mathbf{x}$ is chosen through the key query. Instead, it changes the labels one by one: first it moves the current label value into a hidden IPFE coordinate, then it invokes the piecewise property of AKGS to replace it by a simulated value, and finally it moves the value back into the normal form. This is exactly the "piecemeal" simulation pattern used in the one-key proof of [LL20a], transposed to the ciphertext-policy setting.

From one key to many keys. The one-key construction above uses a single scalar pad μ. In the many-key setting, the same ciphertext must interact consistently with many independently generated keys. We therefore replace μ by a bilinear pad $s \cdot \mu$, where a key contains a random scalar $s \in \mathbb{Z}_p$ and a ciphertext contains $\mu, \eta \in \mathbb{Z}_p$. The ciphertext creates an independent AKGS garbling with secrets

$$(\alpha, \beta) = \begin{cases} (\mu, 0) & \text{for the nonzero mode,} \\ (\eta, \mu) & \text{for the zero mode.} \end{cases}$$

The vectors in the many-key construction can be summarized as follows, where the displayed columns coef_i

range over all $i \in [n]$.

	pad	const	coef_i	$\mathfrak{s}_{\mathrm{pub}}$	$\mathfrak{s}_{\mathrm{priv}}$
$\mathbf{V}_{\mathrm{ip}}$	s	0	0	$\mathbf{v}_{\mathrm{pub}}$	$\mathbf{v}_{\mathrm{priv}}$
$\mathbf{V}_{\mathrm{att}}$	0	s	$s\mathbf{x}[i]$	0	0
$\mathbf{U}_{\mathrm{ip}}$	μ	0	0	$\mathbf{u}_{\mathrm{pub}}$	$\mathbf{u}_{\mathrm{priv}}$
$\mathbf{U}_j$	0	$\mathbf{L}_j[\mathsf{const}]$	$\mathbf{L}_j[\mathsf{coef}_i]$	0	0

Thus the payload copy decrypts to

$$[\![s\mu + \mathbf{v}\mathbf{u}^\top]\!]_{\mathrm{t}},$$

and the label copy returns the randomized labels $s \cdot L_j(\mathbf{x})$. By linearity of AKGS, these randomized labels are themselves a valid garbling whose secrets are the corresponding randomized secrets. Hence the AKGS output recovers $s \cdot \mu$ on authorized pairs, and the decryptor subtracts this value from the payload output.

Adaptive security in the many-key setting follows the same conceptual path as the full ABE proof in [LL20a], but now the proof must also keep the inner-product payload consistent for keys that are authorized for a ciphertext. We handle the key queries one at a time. Fix a key query q and write

$$z_{q,r}^{(b)} = \mathbf{v}_{q,b}\mathbf{u}_{r,b}^\top$$

for the slotted inner product between the q-th queried key vector and the r-th queried ciphertext vector in challenge branch $b \in \{0,1\}$. The one-key label-switching argument is applied simultaneously to all ciphertext queries r, so that the interaction of key q with ciphertext r is described by a payload value of the form $[\![s_q\mu_r + z_{q,r}^{(b)}]\!]_{\mathrm{t}}$ together with AKGS labels that either reconstruct the pad $s_q\mu_r$ or are simulated. Thus, when the proof changes the ciphertext payload vector from $\mathbf{u}_{r,0}$ to $\mathbf{u}_{r,1}$, the ciphertext still interacts correctly with authorized keys: if $f_r(\mathbf{x}_q) = 1$, the AB-sIPFE admissibility condition gives $z_{q,r}^{(0)} = z_{q,r}^{(1)}$, and hence the authorized decryption value is preserved exactly. In other words, authorized entries are carried through the hybrids as fixed target values, rather than being randomized.

For unauthorized entries, the AKGS piecewise simulation hides the label values associated with key q, and then a standard source-group DDH row switch replaces the vector of pad pairings $[\![s_q\mu_r]\!]_{\mathrm{t}}$ associated with the row $\mathbf{s}_q$ by independent target-group pads. After this switch, an unauthorized payload value has the form $\omega_{q,r} + z_{q,r}^{(b)}$ for a fresh pad $\omega_{q,r}$, so changing from $z_{q,r}^{(0)}$ to $z_{q,r}^{(1)}$ is absorbed by the random shift of $\omega_{q,r}$. Note that the q-th hybrid modifies only the row of the implicit decryption table involving the active key, while previously processed rows are already bit-independent and unprocessed rows remain in their original form. Repeating this row-by-row argument over all key queries removes the challenge bit from the adversary's view.

Making the scheme public-key. The last step is to replace the underlying IPFE copies by standard two-slot sIPFE copies. Honest public encryption places the visible input vector in the public slots and sets all private slots to zero. In the security proof, the challenge ciphertext is moved into the private slots, where the function-hiding property of sIPFE allows the same label-switching and MDDH row-switching argument to go through. This is exactly the role played by slotted IPFE in [LL20a]: public slots give a public encryption algorithm, while private slots provide the hidden coordinates needed for the adaptive proof.

2.4 Full Traceability

As described at the end of Section 2.1, the MA-PLBE to MA-TT compiler does not yet yield a scheme satisfying the full traceability notion of Definition 6.1. As in the recent pairing-based traitor tracing schemes of [Zha25, DDP+25], we are only able to establish a *weak* form of mode indistinguishability against decoder boxes, and this weakness manifests as two limitations of the resulting MA-TT scheme. First, tracing succeeds only with some constant probability $\gamma \in (0,1)$, which results in a scheme known as *γ-risky* MA-TT. Second, tracing is only guaranteed against *α-limited* decoders, that is, those whose distinguishing advantage exceeds a fixed constant α. Put together this results in an MA-TT scheme that is (γ,α)-risky limited. In this section, we explain how to remove both restrictions generically, by lifting the riskiness- and limitedness-mitigation compilers of [Zha20, DDP+25] to the multi-authority setting. The resulting chain of transformations is summarized in Figure 1.

A recurring theme is that both compilers act on the *tracing/decoder* layer and treat the underlying scheme as a black-box. They are therefore largely orthogonal to what is actually decentralized in our construction, namely *key*

distribution. Nevertheless, there are genuinely new difficulties arising from the multi-authority nature of tracing: (i) the traceability experiment runs the tracing algorithm once for *every* honest tracing authority $\mathsf{tid}^* \in H_{\mathsf{trace}}$, and combines the resulting sets $\{J_{\mathsf{tid}^*}\}$ by intersecting over an authorized coalition $T' \in \mathbb{T}$ and accusing $\bigcup_{T'} \bigcap_{\mathsf{tid}\in T'} J_{\mathsf{tid}}$; and (ii) authorities cannot coordinate, so any randomness a compiler needs must be derivable identically and independently by each authority. As we explain below, each compiler reduces its multi-authority overhead to exactly one of these two points.

Removing riskiness. We amplify the success probability of tracing via the riskiness mitigation compiler of [Zha20, DDP+25]. Before describing the compiler, we recall the three events that appear in the traceability guarantee. The event GoodDec says that the adversarial decoder D is useful: on a fresh challenge ciphertext encrypting one of the two challenge messages, it predicts the hidden bit with advantage at least the relevant threshold. The event GoodTr says that tracing succeeds in the multi-authority sense: there is an authorized set of honest tracing authorities $T' \in \mathbb{T}$ whose output sets have a common accused user, i.e. $\bigcap_{\mathsf{tid}\in T'} J_{\mathsf{tid}} \neq \emptyset$. Finally, HonAuth is the event in which the corruption pattern still allows honest tracing: for every user, whenever the tracing keys it could assemble (from honest authorities plus all corrupted ones) form an authorized set, they already contain an authorized set of *honest* authorities. Thus, any authorized coalition the adversary embeds for that user contains an authorized honest sub-coalition, which the honest authorities can jointly trace. This generalizes the no-corruption requirement $H_{\mathsf{trace}} = T$ (all tracing authorities honest), which satisfies it trivially. The riskiness bounds below are understood after conditioning on HonAuth. In the τ-fold parallel repetition experiment of the riskiness compiler (described below), GoodTr_j and $\mathsf{GoodDec}_j$ denote GoodTr and GoodDec restricted to the j-th copy.

We now describe the compiler. To raise the riskiness from the constant γ to any target $\gamma' \in (\gamma, 1]$, the compiler runs $\tau = (\gamma'/\gamma) \cdot \omega(\log \lambda)$ independent parallel copies of the scheme—every tracing authority executes its setup τ times—and encrypts under a single uniformly chosen copy j, and hence, the per-authority traced set becomes $J_{\mathsf{tid}} = \bigcup_{j\in[\tau]} J_{\mathsf{tid},j}$. This composes seamlessly with the multi-authority combine rule because intersection is monotone under union: for any fixed copy j,

$$\bigcap_{\mathsf{tid}\in T'} J_{\mathsf{tid}} = \bigcap_{\mathsf{tid}\in T'} \Big(\bigcup_{j'\in[\tau]} J_{\mathsf{tid},j'} \Big) \supseteq \bigcap_{\mathsf{tid}\in T'} J_{\mathsf{tid},j},$$

and hence, a single successfully traced copy already makes the compiled trace succeed. Thus, the compiled good-tracing event contains $\bigcup_j \mathsf{GoodTr}_j$, which is exactly the event whose probability the correlated-instance analysis of [Zha20] lower-bounds by $\gamma' \cdot \Pr[\mathsf{GoodDec}]$. The only multi-authority specific step is the reduction's embedding of the single-instance challenger into one random copy j^*, while the remaining $\tau - 1$ copies are simulated locally. Because each key query is registered in all copies and each corruption is forwarded to all of them, the bookkeeping sets, and hence, HonAuth, H_{trace} and the set J of registered users, coincide across copies, and the centralized coupling argument goes through verbatim. Taking $\gamma' = 1$ removes riskiness entirely.

Removing limitedness. To trace decoders of arbitrary inverse-polynomial advantage we use the limitedness mitigation compiler of [Zha20, DDP+25]. To encrypt a z-bit message, it y-out-of-y secret-shares each of the z columns, encrypts the shares independently, and masks the message with Goldreich-Levin hardcore bits [GL89] of the column-wise XORs. A decoder for the compiled scheme must therefore decrypt all y shares of some column to learn anything, thus even an inverse-polynomial advantage forces a high per-share success rate—enough to invoke the limited tracing guarantee. The Goldreich-Levin extraction and the hybrid argument over the z columns are inherited from [Zha20] verbatim, as the tracing policy $\mathbb{T}$ appears only as a parameter of the underlying encryptions and never interacts with the extraction. The new multi-authority subtlety is phenomenon (i) described above, i.e., the experiment invokes tracing once per honest authority, and if each authority built its derived decoder from fresh Goldreich-Levin randomness, the per-authority sets would be independent and the intersection clause of GoodTr would break for conjunctive policies $\mathbb{T}$. We resolve this without any coordination among authorities by deriving the Goldreich-Levin masking pair *deterministically* from a random oracle evaluated at public inputs such as the public-keys—notably *excluding* the tracing key $\mathsf{msk}_{\mathsf{tid}^*}$ from the input. Every honest authority then constructs the very same canonically derived decoder, presenting the underlying limited tracer with a single decoder exactly as its guarantee requires.

Chaining the two compilers turns the (γ, α)-risky limited MA-TT scheme of Section 2.1 into a fully traceable one. We remark that the extension to the attribute-based setting and to exponentially large user universes via

embedded-identity tracing [GKW19] is obtained by standard, generic techniques and is deferred to Section 6.2.1.

2.5 Towards Public Traceability

Our construction outlined in Sections 2.1 to 2.4 considers a tracing algorithm that takes as input the master secret key of an authority and can therefore only be executed by that authority. Our framework naturally extends to the setting of public traceability, where tracing can be performed using only public keys.

Unfolding the construction in the secret-traceability setting, we observe that the authority's master secret key in the MA-TT scheme is given by the master secret key of a mixed AB-sIPFE scheme (see Section 2.2.2), and that the tracing procedure internally invokes the secret-mode encryption algorithm of this mixed AB-sIPFE scheme. Thus, eliminating the need for secret keys in tracing amounts to enabling public-mode encryption for comparison predicates $(> i)(\cdot)$ in the underlying mixed AB-sIPFE scheme.

In other words, we require an AB-sIPFE scheme for comparison predicates with strong predicate hiding, where the predicate associated with a ciphertext remains hidden even from decrypting parties.[8] Such a primitive can be instantiated in a straightforward manner when ciphertexts grow linearly in the number of users N; for completeness, we provide such a construction in Appendix A. However, this yields an MA-TT scheme whose ciphertexts inherit the same linear dependence on N.

While this establishes the feasibility of MA-TT with public traceability, it leaves open the problem of achieving more compact constructions. To this end, it suffices to construct an AB-sIPFE scheme with smaller ciphertexts. In the single-authority setting, [BW06, Gay16, BCFG17] show that ABE for comparison predicates with strong predicate hiding and ciphertext size $O(\sqrt{N})$ can be obtained by viewing comparisons as quadratic functions and instantiating functional encryption for quadratic functions so as to realize these comparison predicates. This suggests a potential route towards more compact AB-sIPFE: augmenting the construction of [Gay16, BCFG17] with sIPFE functionality. We leave this as an interesting open problem.

3 Preliminaries

3.1 Notational Conventions

Let $\lambda \in \mathbb{N}$ be the security parameter. Throughout the paper, we will suppress λ in subscripts for brevity. A nonnegative function $\varepsilon\colon \mathbb{N} \to \mathbb{R}$ is negligible if $\varepsilon(\lambda) = O(\lambda^{-n})$ for all $n \in \mathbb{N}$. To avoid confusion, we always write vectors $\mathbf{v}$ and matrices $\mathbf{A}$ in boldface and use uppercase letters for the latter. Scalars s are written in italics. Unless otherwise stated, all vectors $\mathbf{v}$ are viewed as row vectors. The corresponding column vector is denoted by $\mathbf{v}^{\mathrm{T}}$.

Algorithms. An algorithm is said to be *efficient* if it runs in probabilistic polynomial time (PPT) in the security parameter. Here, the polynomial runtime refers to all inputs, not only the security parameter. Adversaries always and only receive the security parameter in unary as input, so their running time is polynomial in the security parameter. We write $x \leftarrow A(y)$ to denote that an algorithm A on input y outputs x. When we want to make the randomness r used by $\mathbf{A}$ explicit, we write it as $x \leftarrow A(y; r)$. If A has oracle access to a procedure O, we represent this by superscript A^O.

Security experiments and distributions. Let Exp be an interactive *experiment* that interacts with an algorithm $\mathcal{A}$ (called the *adversary*), depends on the security parameter λ and has binary outcome. We also refer to such objects as *games* or *hybrids*. We let "$\mathsf{Exp}_{\mathcal{A}}(1^\lambda) \to 1$" denote the event that the outcome of running Exp with $\mathcal{A}$ on security parameter λ is 1. For two experiments Exp^0 and Exp^1, we define the distinguishing advantage of $\mathcal{A}$ against the tuple $(\mathsf{Exp}^0, \mathsf{Exp}^1)$ as

$$\mathsf{Adv}_{\mathsf{Exp}^0,\mathsf{Exp}^1,\mathcal{A}}(\lambda) \coloneqq \left|\Pr\left[\mathsf{Exp}^1_{\mathcal{A}}(1^\lambda) \to 1\right] - \Pr\left[\mathsf{Exp}^0_{\mathcal{A}}(1^\lambda) \to 1\right]\right| .$$

We write $\mathsf{Exp}^0 \approx_c \mathsf{Exp}^1$ if the experiments are *computationally indistinguishable*, i.e., their distinguishing advantage is negligible for all efficient adversaries $\mathcal{A}$. We write $\mathsf{Exp}^0 \approx_s \mathsf{Exp}^1$ if the experiments are *statistically indistinguishable*, i.e., their distinguishing advantage is negligible for all (even unbounded) adversaries. We write $\mathsf{Exp}^0 \equiv \mathsf{Exp}^1$ if the experiments are *identically distributed*, i.e., their distinguishing advantage is 0 for all (even unbounded) adversaries. By default, the term *indistinguishable* refers to computational indistinguishability.

[8] In the PLBE setting, this notion corresponds to index hiding.

More general, the same notations can be used for sequences of distributions. Let $D^0 = \{D^0_\lambda\}_{\lambda\in\mathbb{N}}$ and $D^1 = \{D^1_\lambda\}_{\lambda\in\mathbb{N}}$ be two sequences of distributions. For $b \in \{0,1\}$, we define $\mathsf{Exp}^b_{\mathcal{A}}(1^\lambda)$ as follows: sample $x \leftarrow D^b_\lambda$, run $\mathcal{A}(1^\lambda, x)$ and use the output of $\mathcal{A}$ as the outcome of the experiment. Then we write $D^0 \approx_c D^1$ (resp. $D^0 \approx_s D^1$, $D^0 \equiv D^1$) if $\mathsf{Exp}^0_{\mathcal{A}} \approx_c \mathsf{Exp}^1_{\mathcal{A}}$ (resp. $\mathsf{Exp}^0_{\mathcal{A}} \approx_s \mathsf{Exp}^1_{\mathcal{A}}$, $\mathsf{Exp}^0_{\mathcal{A}} \equiv \mathsf{Exp}^1_{\mathcal{A}}$).

Given a finite set S, we denote by $\mathsf{U}(S)$ the uniform distribution on S and write $x \leftarrow\!\$\, S$ to indicate that the variable x is sampled from $\mathsf{U}(S)$.

Sets and indexing. We denote by $\mathbb{Z}$ and $\mathbb{N}$ the sets of integers and natural numbers (positive integers). For integers m and n, we write $[m;n]$ to denote the set $\{z \in \mathbb{Z} : m \le z \le n\}$ and let $[n] \coloneqq [1;n]$. For a prime number p, $\mathbb{Z}_p$ denotes the finite field of integers modulo p. For a finite set S, we let 2^S denote the power set of S. Furthermore, we denote the disjoint union of two sets S_1, S_2 (not necessarily disjoint) by $S_1 \cup S_2$.

To index a vector, we write $\mathbf{v}[i]$. In contrast, objects of some collection that is not regarded as a vector are indexed using subscripts (or superscripts in some cases). For instance, $\mathbf{v}_i$ represents a vector, not a component of some vector. If i runs through some index set $[n]$, it means that there are n vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$. If the n objects are scalars (or not explicitly vectors), we will write $v_1, \ldots, v_n$ instead.

For convenience, objects might be indexed by arbitrary sets, not just integers. For finite sets $\mathfrak{s}, A$, we write $A^{\mathfrak{s}} \coloneqq \{(\mathbf{v}[i])_{i\in\mathfrak{s}} : \mathbf{v}[i] \in A\}$ for the set of vectors whose entries are in A and indexed by $\mathfrak{s}$, e.g., $\mathbb{Z}_p^{[n]}$ is just $\mathbb{Z}_p^n$. Suppose $\mathfrak{s}_1, \mathfrak{s}_2$ are two index sets with $\mathfrak{s}_1 \subseteq \mathfrak{s}_2$. For a vector $\mathbf{v} \in \mathbb{Z}_p^{\mathfrak{s}_2}$, we denote by $\mathbf{u} = \mathbf{v}|_{\mathfrak{s}_1}$ its canonical projection onto $\mathbb{Z}_p^{\mathfrak{s}_1}$, i.e., $\mathbf{u} \in \mathbb{Z}_p^{\mathfrak{s}_1}$ and $\mathbf{u}[i] = \mathbf{v}[i]$ for all $i \in \mathfrak{s}_1$. Conversely, for any vector $\mathbf{u} \in \mathbb{Z}_p^{\mathfrak{s}_1}$, we write $\mathbf{v} = \mathbf{u}|^{\mathfrak{s}_2}$ for its zero-extension into $\mathbb{Z}_p^{\mathfrak{s}_2}$, i.e., $\mathbf{v} \in \mathbb{Z}_p^{\mathfrak{s}_2}$ and $\mathbf{v}[i] = \mathbf{u}[i]$ if $i \in \mathfrak{s}_1$, and $\mathbf{v}[i] = 0$ if $i \in \mathfrak{s}_2 \setminus \mathfrak{s}_1$.

3.2 Pairing Groups and Hardness Assumptions

Our constructions use a sequence of pairing groups

$$\mathbb{G} = \{\mathbb{G}_\lambda = (\mathbb{G}_{\lambda,1}, \mathbb{G}_{\lambda,2}, \mathbb{G}_{\lambda,\mathrm{t}}, g_{\lambda,1}, g_{\lambda,2}, g_{\lambda,\mathrm{t}}, e_\lambda, p_\lambda)\}_{\lambda\in\mathbb{N}}\,,$$

where $\mathbb{G}_{\lambda,1}$ (resp. $\mathbb{G}_{\lambda,2}$, $\mathbb{G}_{\lambda,\mathrm{t}}$) is a cyclic group of order p_λ generated by $g_{\lambda,1}$ (resp. $g_{\lambda,2}$, $g_{\lambda,\mathrm{t}}$), and $e_\lambda : \mathbb{G}_{\lambda,1}\times\mathbb{G}_{\lambda,2} \to \mathbb{G}_{\lambda,\mathrm{t}}$ is the pairing operation satisfying $e_\lambda(g^a_{\lambda,1}, g^b_{\lambda,2}) = g^{ab}_{\lambda,\mathrm{t}}$ for all integers a, b. The group operations and the pairing map are required to be efficiently computable.

Following the implicit notation in [EHK+13], we write $[\![a]\!]_i$ to denote $g^a_{\lambda,i}$ for $i \in \{1,2,\mathrm{t}\}$. This notation extends component-wise to matrices and vectors having entries in $\mathbb{Z}_p$. Equipped with these notations, group operations are written additively and the pairing operation multiplicatively, *e.g.* $[\![\mathbf{A}]\!]_1 - \mathbf{B}[\![\mathbf{C}]\!]_1\mathbf{D} = [\![\mathbf{A} - \mathbf{BCD}]\!]_1$ and $[\![\mathbf{A}]\!]_1[\![\mathbf{B}]\!]_2 = [\![\mathbf{AB}]\!]_\mathrm{t}$.

Next we recall the MDDH assumption defined in [EHK+13].

Assumption 3.1 ($\mathsf{MDDH}^q_{k,\ell}$). Let $\alpha \in \{1,2,\mathrm{t}\}$, $k \ge 1$ an integer constant and $\ell = \ell(\lambda), q = q(\lambda)$ some polynomials. For a sequence of pairing groups $\mathbb{G}$, the $\mathsf{MDDH}^q_{k,\ell}$ assumption holds in $\mathbb{G}_\alpha$ if $\{[\![\mathbf{A}]\!]_\alpha, [\![\mathbf{AS}]\!]_\alpha\}_{\lambda\in\mathbb{N}} \approx_c \{[\![\mathbf{A}]\!]_\alpha, [\![\mathbf{U}]\!]_\alpha\}_{\lambda\in\mathbb{N}}$ for $\mathbf{A} \leftarrow\!\$\, \mathbb{Z}_p^{\ell\times k}$, $\mathbf{S} \leftarrow\!\$\, \mathbb{Z}_p^{k\times q}$ and $\mathbf{U} \leftarrow\!\$\, \mathbb{Z}_p^{\ell\times q}$.

In [EHK+13], it was shown that k-Lin implies $\mathsf{MDDH}^1_{k,k+1}$ which in turn implies $\mathsf{MDDH}^q_{k,\ell}$ for all $\ell > k$ and $q \ge 1$ with a tight security reduction. For convenience, we sometimes refer to the $\mathsf{MDDH}^1_{k,k+1}$ assumption by MDDH_k.

3.3 Monotone Access Structures and Linear Secret Sharing Schemes

We recall the definitions of monotone access structures.

Definition 3.2 (Monotone Access Structures, [BL90, Bei96]). Let A be a set. An access structure over A is a collection $\mathbb{A} \subseteq 2^A$ of subsets of A. We say that a set $S \subseteq A$ is *authorized* if $S \in \mathbb{A}$. We say that the access structure $\mathbb{A}$ is *monotone* if for all $S_1 \subseteq S_2$, whenever $S_1 \in \mathbb{A}$ then it holds that $S_2 \in \mathbb{A}$.

Next, we recall the notion of a linear secret sharing scheme for an access structure $\mathbb{A}$.

Definition 3.3 (Linear secret sharing scheme [BL90, Bei96, LW11]). Let A be a set and $\mathbb{A}$ an access policy over A. A linear secret sharing scheme for $\mathbb{A}$ over $\mathbb{Z}_p$ is a pair $(\mathbf{M}, \rho)$, where $\mathbf{M} \in \mathbb{Z}_p^{n\times\ell}$ is the share-generating matrix and $\rho\colon [\ell] \to A$ is a row-labeling function. The pair $(\mathbf{M}, \rho)$ should satisfy the following properties:

- **Share generation:** To share a secret $\sigma \in \mathbb{Z}_p$, sample $\mathbf{r} \leftarrow\!\!\$\ \mathbb{Z}_p^{n-1}$ and define the vector $\mathbf{u} = (\sigma, \mathbf{r})$. Then $(\sigma_1, \ldots, \sigma_\ell) = \mathbf{uM}$ is the vector of shares where $\sigma_i \in \mathbb{Z}_p$ belongs to party $\rho(i)$ for each $i \in [\ell]$.
- **Share reconstruction:** For every $S \subseteq A$ such that $S \in \mathbb{A}$, there exists a vector $\mathbf{c} = (c_1, \ldots, c_\ell) \in \mathbb{Z}_p^\ell$ where $c_i = 0$ whenever $\rho(i) \notin S$ such that $\mathbf{M} \cdot \mathbf{c}^\mathsf{T} = \mathbf{e}_1^\top$. Here, $\mathbf{e}_1 = (1, 0, \ldots, 0) \in \mathbb{Z}_p^n$ is the first elementary basis vector.
- **Security:** For every $S \subseteq A$ such that $S \notin \mathbb{A}$, we have $\mathbf{e}_1 \notin \operatorname{span} \mathbf{M}_J$, where $J \coloneqq \rho^{-1}(S)$ and $\mathbf{M}_J$ denotes the matrix formed by taking the subset of columns in $\mathbf{M}$ indexed by elements in J.

The work of [BL90] gives an implicit linear secret sharing scheme over any field $\mathbb{Z}_p$ for monotone access structures. We refer to [LW11, Appendix G] for an explicit description of the share-generating matrix.

Definition 3.4 (One-use LSSS). Let $\mathbb{A} \subseteq 2^S$ be an access structure with an associated linear secret sharing scheme $(\mathbf{M}, \rho)$ over $\mathbb{Z}_p$. We say that $(\mathbf{M}, \rho)$ is a one-use linear secret sharing scheme if the function ρ is injective.

In this work, we make use of one-use LSSS.

3.4 Slotted Inner-Product Functional Encryption

We recall the notion of slotted IPFE [LL20a].

Definition 3.5 (Slotted IPFE). Let $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, e, p)$ be a pairing group. A slotted inner-product functional encryption (sIPFE) scheme based on $\mathbb{G}$ consists of five procedures whose syntax is given below.

- $\mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathrm{pub}}, \mathfrak{s}_{\mathrm{priv}}) \to (\mathsf{mpk}, \mathsf{msk})$: On input the security parameter 1^λ and two disjoint index sets, the public slots $\mathfrak{s}_{\mathrm{pub}}$ and the private slots $\mathfrak{s}_{\mathrm{priv}}$, it outputs a pair of master public and master secret key $(\mathsf{mpk}, \mathsf{msk})$. The whole index set $\mathfrak{s}$ is $\mathfrak{s}_{\mathrm{pub}} \cup \mathfrak{s}_{\mathrm{priv}}$.
- $\mathsf{KeyGen}(\mathsf{msk}, [\![\mathbf{v}]\!]_2) \to \mathsf{sk}$: On input the master secret key msk and a function vector $\mathbf{v} \in \mathbb{Z}_p^{\mathfrak{s}}$ encoded in $\mathbb{G}_2$, it outputs a secret key sk.
- $\mathsf{EncPK}(\mathsf{mpk}, [\![\mathbf{u}_{\mathrm{pub}}]\!]_1) \to \mathsf{ct}$: On input the master public key mpk and a message vector $\mathbf{u}_{\mathrm{pub}} \in \mathbb{Z}_p^{\mathfrak{s}_{\mathrm{pub}}}$ encoded in $\mathbb{G}_1$, it outputs a ciphertext ct.
- $\mathsf{EncSK}(\mathsf{msk}, [\![\mathbf{u}]\!]_1) \to \mathsf{ct}$: On input the master secret key msk and a message vector $\mathbf{u} \in \mathbb{Z}_p^{\mathfrak{s}}$ encoded in $\mathbb{G}_1$, it outputs a ciphertext ct.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to [\![d]\!]_\mathsf{t} \vee \perp$: On input a secret key sk and a ciphertext ct, it outputs a value $[\![d]\!]_\mathsf{t} \in \mathbb{G}_\mathsf{t}$.

Moreover, an sIPFE scheme Π should satisfy the following properties.

- **Correctness:** We say that Π is correct if for all $\lambda \in \mathbb{N}$, all disjoint index sets $\mathfrak{s}_{\mathrm{pub}}, \mathfrak{s}_{\mathrm{priv}}$, and all vectors $\mathbf{u}, \mathbf{v} \in \mathbb{Z}_p^{\mathfrak{s}}$, it holds that

$$\Pr\left[[\![d]\!]_\mathsf{t} = [\![\mathbf{v} \cdot \mathbf{u}^\mathsf{T}]\!]_\mathsf{t} \;\middle|\; \begin{array}{l} (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathrm{pub}}, \mathfrak{s}_{\mathrm{priv}}) \\ \mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, [\![\mathbf{v}]\!]_2) \\ \mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{msk}, [\![\mathbf{u}]\!]_1) \\ [\![d]\!]_\mathsf{t} \coloneqq \mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \end{array} \right] = 1.$$

- **Mode indistinguishability:** We say that Π is mode indistinguishable if for all $\lambda \in \mathbb{N}$, all disjoint index sets $\mathfrak{s}_{\mathrm{pub}}, \mathfrak{s}_{\mathrm{priv}}$, and all vectors $\mathbf{u}_{\mathrm{pub}} \in \mathbb{Z}_p^{\mathfrak{s}_{\mathrm{pub}}}$, the following distributions are identical:

$$\left\{ (\mathsf{mpk}, \mathsf{msk}, \mathsf{ct}) \;\middle|\; \begin{array}{l} (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathrm{pub}}, \mathfrak{s}_{\mathrm{priv}}) \\ \mathsf{ct} \leftarrow \mathsf{EncPK}(\mathsf{mpk}, [\![\mathbf{u}_{\mathrm{pub}}]\!]_1) \end{array} \right\}$$
$$\left\{ (\mathsf{mpk}, \mathsf{msk}, \mathsf{ct}) \;\middle|\; \begin{array}{l} (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathrm{pub}}, \mathfrak{s}_{\mathrm{priv}}) \\ \mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{msk}, [\![\mathbf{u}_{\mathrm{pub}}|^{\mathfrak{s}}]\!]_1) \end{array} \right\}.$$

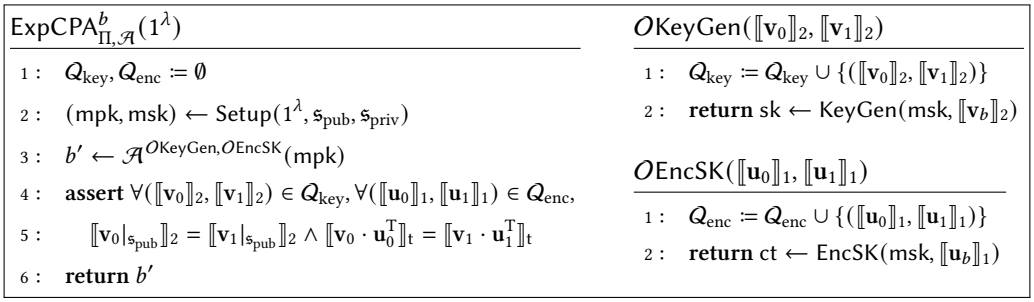

Figure 2: Experiment $\mathsf{ExpCPA}^{b}_{\Pi,\mathcal{A}}$ for Definition 3.5

- **IND-CPA security:** We say that Π is IND-CPA secure if for every stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such for all $\lambda \in \mathbb{N}$,

$$\left|\Pr\left[\mathsf{ExpCPA}^{1}_{\Pi,\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{ExpCPA}^{0}_{\Pi,\mathcal{A}}(1^\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda),$$

  where the experiment $\mathsf{ExpCPA}^{b}_{\Pi,\mathcal{A}}(\lambda)$ is defined in Figure 2.

We call Π an sIPFE scheme if $\mathcal{F}$ is a singleton containing only the constant function that outputs 1 on every input. For brevity, we omit the inputs x and f to the algorithms KeyGen, EncPK and EncSK in this case.

3.5 Mixed Functional Encryption

We recall the definition of a mixed functional encryption scheme [GKW18, Zha25].

Definition 3.6 (MFE). A mixed functional encryption (MFE) scheme for a function class $\mathcal{F} : \mathcal{X} \to \{0,1\}$ consists of five procedures whose syntax is given below.

- $\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$: On input the security parameter 1^λ, it outputs a pair of master public and master secret key $(\mathsf{mpk}, \mathsf{msk})$.
- $\mathsf{KeyGen}(\mathsf{msk}, x) \to \mathsf{sk}$: On input the master secret key msk and an attribute $x \in \mathcal{X}$, it outputs a secret key sk.
- $\mathsf{EncPK}(\mathsf{mpk}) \to \mathsf{ct}$: On input the master public key mpk, it outputs a ciphertext ct.
- $\mathsf{EncSK}(\mathsf{msk}, f) \to \mathsf{ct}$: On input the master secret key msk and a function $f \in \mathcal{F}$, it outputs a ciphertext ct.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to d$: On input a secret key sk and a ciphertext ct, it outputs a single bit $d \in \{0,1\}$.

Moreover, an MFE scheme Π should satisfy the following properties.

- **Correctness:** We say that Π is correct if there exist negligible functions $\mathsf{negl}_{\text{pub}}, \mathsf{negl}_{\text{priv}}$ such that for all $\lambda \in \mathbb{N}$, all $x \in \mathcal{X}$, and all $f \in \mathcal{F}$, that

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = 1 \;\middle|\; \begin{array}{l}(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)\\ \mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, x)\\ \mathsf{ct} \leftarrow \mathsf{EncPK}(\mathsf{mpk})\end{array}\right] \geq 1 - \mathsf{negl}_{\text{pub}}(\lambda),$$

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = f(x) \;\middle|\; \begin{array}{l}(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)\\ \mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, x)\\ \mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{msk}, f)\end{array}\right] \geq 1 - \mathsf{negl}_{\text{priv}}(\lambda).$$

- **Mode indistinguishability:** Let $q(\cdot)$ be any polynomial. We say that Π is q-bounded mode indistinguishable if for every stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\left|\Pr\left[\mathsf{ExpMode}^{q,1}_{\Pi,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{ExpMode}^{q,0}_{\Pi,\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda),$$

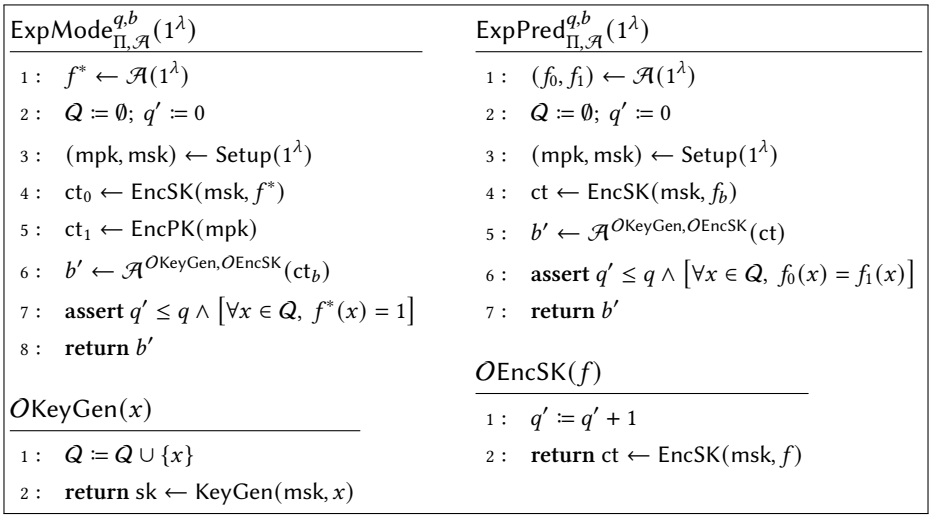

Figure 3: Experiments $\mathsf{ExpMode}^{q,b}_{\Pi,\mathcal{A}}$ and $\mathsf{ExpPred}^{q,b}_{\Pi,\mathcal{A}}$ for Definition 3.6

where the experiment $\mathsf{ExpMode}^{q,b}_{\Pi,\mathcal{A}}$ is defined in Figure 3.

- **Function hiding:** Let $q(\cdot)$ be any polynomial. We say that Π is q-bounded function hiding if for every stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\left|\Pr\left[\mathsf{ExpPred}^{q,1}_{\Pi,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{ExpPred}^{q,0}_{\Pi,\mathcal{A}}(\lambda) = 1\right]\right| \le \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{ExpPred}^{q,b}_{\Pi,\mathcal{A}}$ is defined in Figure 3.

4 Attribute-Based Slotted Inner-Product Functional Encryption

4.1 Definition

We extend the syntax of slotted IPFE to the attribute-based setting and define several security requirements.

Definition 4.1 (AB-sIPFE). Let $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, e, p)$ be a pairing group. An attribute-based slotted inner-product functional encryption (AB-sIPFE) scheme based on $\mathbb{G}$ for a function class $\mathcal{F} : \mathcal{X} \to \{0,1\}$ consists of five procedures whose syntax is given below.

- $\mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathrm{pub}}, \mathfrak{s}_{\mathrm{priv}}) \to (\mathsf{mpk}, \mathsf{msk})$: On input the security parameter 1^λ and two disjoint index sets, the public slots $\mathfrak{s}_{\mathrm{pub}}$ and the private slots $\mathfrak{s}_{\mathrm{priv}}$, it outputs a pair of master public and master secret key $(\mathsf{mpk}, \mathsf{msk})$. The whole index set $\mathfrak{s}$ is $\mathfrak{s}_{\mathrm{pub}} \cup \mathfrak{s}_{\mathrm{priv}}$.
- $\mathsf{KeyGen}(\mathsf{msk}, x, [\![\mathbf{v}]\!]_2) \to \mathsf{sk}$: On input the master secret key msk, an attribute $x \in \mathcal{X}$, and a function vector $\mathbf{v} \in \mathbb{Z}_p^{\mathfrak{s}}$ encoded in $\mathbb{G}_2$, it outputs a secret key sk.
- $\mathsf{EncPK}(\mathsf{mpk}, f, [\![\mathbf{u}_{\mathrm{pub}}]\!]_1) \to \mathsf{ct}$: On input the master public key mpk, a function $f \in \mathcal{F}$ and a message vector $\mathbf{u}_{\mathrm{pub}} \in \mathbb{Z}_p^{\mathfrak{s}_{\mathrm{pub}}}$ encoded in $\mathbb{G}_1$, it outputs a ciphertext ct.
- $\mathsf{EncSK}(\mathsf{msk}, f, [\![\mathbf{u}]\!]_1) \to \mathsf{ct}$: On input the master secret key msk, a function $f \in \mathcal{F}$, and a message vector $\mathbf{u} \in \mathbb{Z}_p^{\mathfrak{s}}$ encoded in $\mathbb{G}_1$, it outputs a ciphertext ct.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to [\![d]\!]_t \vee \perp$: On input a secret key sk and a ciphertext ct, it outputs a value $[\![d]\!]_t \in \mathbb{G}_t$ or $\perp$ (a designated error symbol).

Moreover, an AB-sIPFE scheme Π should satisfy the following properties.

- **Correctness:** We say that Π is correct if for all $\lambda \in \mathbb{N}$, all disjoint index sets $\mathfrak{s}_{\mathrm{pub}}, \mathfrak{s}_{\mathrm{priv}}$, all vectors $\mathbf{u}_{\mathrm{pub}} \in \mathbb{Z}_p^{\mathfrak{s}_{\mathrm{pub}}}$, $\mathbf{u}, \mathbf{v} \in \mathbb{Z}_p^{\mathfrak{s}}$, all $x \in \mathcal{X}$, and all $f \in \mathcal{F}$ such that $f(x) = 1$, there exist negligible functions $\mathsf{negl}_{\mathrm{pub}}, \mathsf{negl}_{\mathrm{priv}}$ such

that

$$\Pr\left[\mathsf{Dec}(\mathsf{sk},\mathsf{ct}) = [\![\mathbf{v}\cdot(\mathbf{u}_{\mathrm{pub}}|^{\mathfrak{s}})^{\mathrm{T}}]\!]_{\mathrm{t}} \;\middle|\; \begin{array}{l}(\mathsf{mpk},\mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda,\mathfrak{s}_{\mathrm{pub}},\mathfrak{s}_{\mathrm{priv}})\\ \mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{msk},x,[\![\mathbf{v}]\!]_2)\\ \mathsf{ct} \leftarrow \mathsf{EncPK}(\mathsf{mpk},f,[\![\mathbf{u}_{\mathrm{pub}}]\!]_1)\end{array}\right] \geq 1-\mathsf{negl}_{\mathrm{pub}}(\lambda),$$

$$\Pr\left[\mathsf{Dec}(\mathsf{sk},\mathsf{ct}) = [\![\mathbf{v}\cdot\mathbf{u}^{\mathrm{T}}]\!]_{\mathrm{t}} \;\middle|\; \begin{array}{l}(\mathsf{mpk},\mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda,\mathfrak{s}_{\mathrm{pub}},\mathfrak{s}_{\mathrm{priv}})\\ \mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{msk},x,[\![\mathbf{v}]\!]_2)\\ \mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{msk},f,[\![\mathbf{u}]\!]_1)\end{array}\right] \geq 1-\mathsf{negl}_{\mathrm{priv}}(\lambda).$$

- **Mode indistinguishability:** Let $q(\cdot)$ be any polynomial. We say that Π is q-bounded mode indistinguishable[9] if for every stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\left|\Pr\left[\mathsf{ExpMode}^{q,1}_{\Pi,\mathcal{A}}(1^\lambda)=1\right]-\Pr\left[\mathsf{ExpMode}^{q,0}_{\Pi,\mathcal{A}}(1^\lambda)=1\right]\right| \leq \mathsf{negl}(\lambda),$$

  where the experiment $\mathsf{ExpMode}^{q,b}_{\Pi,\mathcal{A}}$ is defined in Figure 4.
- **Predicate hiding:** Let $q(\cdot)$ be any polynomial. We say that Π is q-bounded predicate hiding if for every stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\left|\Pr\left[\mathsf{ExpPred}^{q,1}_{\Pi,\mathcal{A}}(1^\lambda)=1\right]-\Pr\left[\mathsf{ExpPred}^{q,0}_{\Pi,\mathcal{A}}(1^\lambda)=1\right]\right| \leq \mathsf{negl}(\lambda),$$

  where the experiment $\mathsf{ExpPred}^{q,b}_{\Pi,\mathcal{A}}$ is defined in Figure 4.
- **Message hiding:** Let $q(\cdot)$ be any polynomial. We say that Π is q-bounded message hiding if for every stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\left|\Pr\left[\mathsf{ExpMsg}^{q,1}_{\Pi,\mathcal{A}}(1^\lambda)=1\right]-\Pr\left[\mathsf{ExpMsg}^{q,0}_{\Pi,\mathcal{A}}(1^\lambda)=1\right]\right| \leq \mathsf{negl}(\lambda),$$

  where the experiment $\mathsf{ExpMsg}^{q,b}_{\Pi,\mathcal{A}}$ is defined in Figure 4.

Some of our constructions support public-mode encryption only for the constant function $\mathbf{1}$, which evaluates to 1 on every input. Such schemes can be viewed as an extension of (message-carrying) mixed functional encryption [GKW18, Zha25] with the additional sIPFE functionality. Motivated by this connection, we define the following notion of *mixed* AB-sIPFE.

Definition 4.2 (Mixed AB-sIPFE). An AB-sIPFE scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{EncPK}, \mathsf{EncSK}, \mathsf{Dec})$ for a function class $\mathcal{F}$ is called mixed if the public-mode encryption algorithm EncPK is defined only for the constant function $\mathbf{1} \in \mathcal{F}$. That is, $\mathsf{EncPK}(\mathsf{mpk}, f, [\![\mathbf{u}_{\mathrm{pub}}]\!]_1)$ is supported if and only if $f = \mathbf{1}$.

4.2 Construction of AB-sIPFE for Arithmetic Branching Programs

This section constructs the AB-sIPFE primitive of Definition 4.1 for ciphertext policies represented by arithmetic branching programs (ABPs). The construction is the ciphertext-policy analogue of the AKGS/sIPFE framework of [LL20a]. We first describe the arithmetic key-garbling (AKGS) tool and the one-key core, and then lift the core to a public-key, many-key AB-sIPFE using a standard two-slot sIPFE.

A policy is a pair

$$\pi = (f, \mathsf{mode}), \qquad \mathsf{mode} \in \{\mathsf{nz}, \mathsf{zero}\},$$

where $f : \mathbb{Z}_p^n \to \mathbb{Z}_p$ is an ABP. It represents the Boolean predicate

$$\pi(\mathbf{x}) = 1 \iff \begin{cases} f(\mathbf{x}) \neq 0, & \mathsf{mode} = \mathsf{nz},\\ f(\mathbf{x}) = 0, & \mathsf{mode} = \mathsf{zero}. \end{cases}$$

[9] In the literature, this property is also called *slot-mode correctness* [LL20a], *normal hiding security* [GKW18] or *accept indistinguishability* [Zha25]. We prefer to use the term mode indistinguishability here to remain consistent with other definitions in this work.

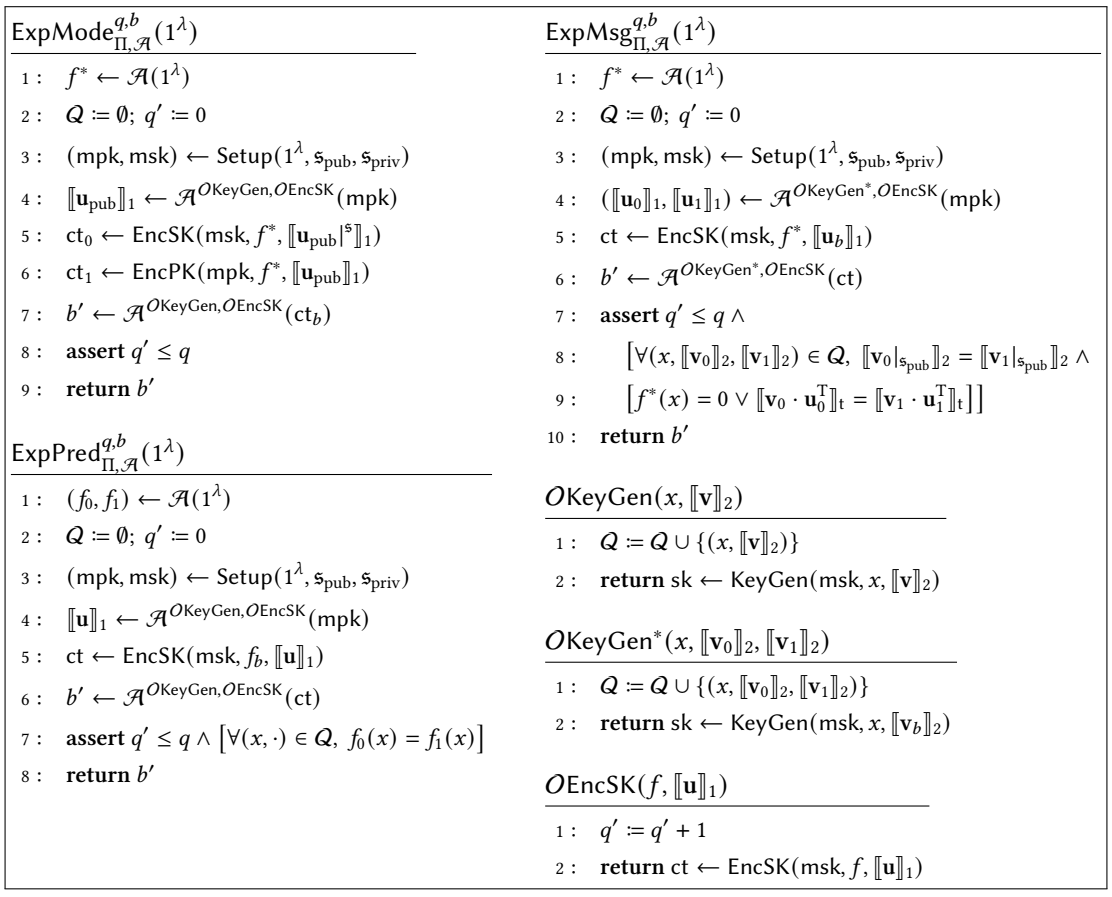

Figure 4: Experiments $\mathsf{ExpMode}^{q,b}_{\Pi,\mathcal{A}}$, $\mathsf{ExpPred}^{q,b}_{\Pi,\mathcal{A}}$ and $\mathsf{ExpMsg}^{q,b}_{\Pi,\mathcal{A}}$ for Definition 4.1

$\mathsf{ExpCPA}^{b}_{\Pi,\mathcal{A}}(1^\lambda)$

$Q_{\mathrm{key}}, Q_{\mathrm{enc}} \coloneqq \emptyset$
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathbb{s}_{\mathrm{pub}}, \mathbb{s}_{\mathrm{priv}})$
$b' \leftarrow \mathcal{A}^{O\mathsf{KeyGen}, O\mathsf{EncSK}}(\mathsf{mpk})$
assert $\forall (x, [\![\mathbf{v}_0]\!]_2, [\![\mathbf{v}_1]\!]_2) \in Q_{\mathrm{key}}, \forall (f, [\![\mathbf{u}_0]\!]_1, [\![\mathbf{u}_1]\!]_1) \in Q_{\mathrm{enc}},$
  $[\![\mathbf{v}_0|_{\mathbb{s}_{\mathrm{pub}}}]\!]_2 = [\![\mathbf{v}_1|_{\mathbb{s}_{\mathrm{pub}}}]\!]_2 \wedge \left[f(x) = 0 \vee [\![\mathbf{v}_0 \cdot \mathbf{u}_0^{\mathrm{T}}]\!]_{\mathrm{t}} = [\![\mathbf{v}_1 \cdot \mathbf{u}_1^{\mathrm{T}}]\!]_{\mathrm{t}} \right]$
return b'

$O\mathsf{KeyGen}(x, [\![\mathbf{v}_0]\!]_2, [\![\mathbf{v}_1]\!]_2)$

$Q_{\mathrm{key}} \coloneqq Q_{\mathrm{key}} \cup \{(x, [\![\mathbf{v}_0]\!]_2, [\![\mathbf{v}_1]\!]_2)\}$
return $\mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, x, [\![\mathbf{v}_b]\!]_2)$

$O\mathsf{EncSK}(f, [\![\mathbf{u}_0]\!]_1, [\![\mathbf{u}_1]\!]_1)$

$Q_{\mathrm{enc}} \coloneqq Q_{\mathrm{enc}} \cup \{(f, [\![\mathbf{u}_0]\!]_1, [\![\mathbf{u}_1]\!]_1)\}$
return $\mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{msk}, f, [\![\mathbf{u}_b]\!]_1)$

Figure 5: Experiment $\mathsf{ExpCPA}^{b}_{\Pi,\mathcal{A}}$

Secret keys are associated with attributes $\mathbf{x} \in \mathbb{Z}_p^n$ and vectors $\mathbf{v} \in \mathbb{Z}_p^{\mathbb{s}_{\mathrm{pub}} \cup \mathbb{s}_{\mathrm{priv}}}$, ciphertexts are associated with policies π and vectors $\mathbf{u} \in \mathbb{Z}_p^{\mathbb{s}_{\mathrm{pub}} \cup \mathbb{s}_{\mathrm{priv}}}$, and decryption returns $[\![\mathbf{v}\mathbf{u}^{\mathrm{T}}]\!]_{\mathrm{t}}$ exactly when $\pi(\mathbf{x}) = 1$.

Unbounded and adaptive security. We believe that our AB-sIPFE construction for ABPs may find applications beyond traitor tracing. We therefore consider a stronger IND-CPA security notion which is not required for the results of this work but may prove useful in other contexts. The experiment $\mathsf{ExpCPA}^{b}_{\Pi,\mathcal{A}}$ defined in Figure 5 constitutes a strict strengthening of the message-hiding property from Definition 4.1. Specifically, unlike q-bounded message hiding, it places no bound on the number of $O\mathsf{EncSK}$ queries (equivalently, it corresponds to the case where q is unbounded), and it is fully adaptive, as the adversary is not required to commit to the challenge policy f^* at the outset of the game.

4.2.1 Arithmetic Key Garbling for ABPs

We begin with the arithmetic key-garbling tool. Throughout this subsection, n is the input length, $\mathbf{x} = (\mathbf{x}[1], \ldots, \mathbf{x}[n])$ is a formal row vector of variables, and

$$\overline{\mathbf{x}} \coloneqq (1, \mathbf{x}[1], \ldots, \mathbf{x}[n]) \in \mathbb{Z}_p^{n+1}.$$

A label function L_j is represented by a coefficient row vector $\mathbf{L}_j \in \mathbb{Z}_p^{n+1}$ and is evaluated as

$$L_j(\mathbf{x}) = \mathbf{L}_j \overline{\mathbf{x}}^{\mathrm{T}} = \mathbf{L}_j[\mathsf{const}] + \sum_{i \in [n]} \mathbf{L}_j[\mathsf{coef}_i]\mathbf{x}[i].$$

When there are m labels, we write $\mathbf{L} \in \mathbb{Z}_p^{m \times (n+1)}$ for the matrix whose j-th row is $\mathbf{L}_j$, and $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_m) \in \mathbb{Z}_p^m$ for the row vector of label values.

Definition 4.3 (Linear AKGS with special piecewise simulation). A linear arithmetic key-garbling scheme for ABPs consists of algorithms Garble, Eval and SimAKGS satisfying the following syntax and properties.

- $\mathsf{Garble}(f, \alpha, \beta; \mathbf{r}) \to (L_1, \ldots, L_m)$: On input an ABP $f : \mathbb{Z}_p^n \to \mathbb{Z}_p$, two scalars $\alpha, \beta \in \mathbb{Z}_p$, and a randomness row vector $\mathbf{r} \in \mathbb{Z}_p^{\rho(f)}$, it outputs a tuple of coefficient row vectors $\mathbf{L}_j \in \mathbb{Z}_p^{n+1}$, $j \in [m]$, where $m = m(f)$ depends only on the ABP-representation of f. Equivalently, the output is a matrix $\mathbf{L} \in \mathbb{Z}_p^{m \times (n+1)}$.
- $\mathsf{Eval}(f, \mathbf{x}, \ell_1, \ldots, \ell_m) \to \gamma$: On input an ABP $f : \mathbb{Z}_p^n \to \mathbb{Z}_p$, an attribute vector $\mathbf{x} \in \mathbb{Z}_p^n$, and a label row vector $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_m) \in \mathbb{Z}_p^m$, it outputs a scalar $\gamma \in \mathbb{Z}_p$. We require evaluation to be linear in the labels: for every f and $\mathbf{x}$, there is an efficiently computable row vector $\mathbf{a}_{f,\mathbf{x}} \in \mathbb{Z}_p^m$ such that

  $$\mathsf{Eval}(f, \mathbf{x}, \ell_1, \ldots, \ell_m) = \mathbf{a}_{f,\mathbf{x}} \boldsymbol{\ell}^{\mathrm{T}}.$$

  Consequently, target-group encodings of the labels can be evaluated in the exponent:

  $$\mathsf{Eval}^{\mathrm{exp}}(f, \mathbf{x}, [\![\ell_1]\!]_{\mathrm{t}}, \ldots, [\![\ell_m]\!]_{\mathrm{t}}) = [\![\mathsf{Eval}(f, \mathbf{x}, \ell_1, \ldots, \ell_m)]\!]_{\mathrm{t}}.$$

- **Correctness:** For every $f, \alpha, \beta, \mathbf{r}$ and $\mathbf{x}$, if $(L_1, \ldots, L_m) \leftarrow \mathsf{Garble}(f, \alpha, \beta; \mathbf{r})$ and $\ell_j = \mathbf{L}_j \overline{\mathbf{x}}^{\mathrm{T}}$ for all $j \in [m]$,

then

$$\mathsf{Eval}(f, \mathbf{x}, \ell_1, \ldots, \ell_m) = \alpha f(\mathbf{x}) + \beta.$$

- **Linearity:** For every fixed f, the matrix $\mathbf{L} \in \mathbb{Z}_p^{m\times(n+1)}$ output by $\mathsf{Garble}(f, \alpha, \beta; \mathbf{r})$ is an affine-linear function of $(\alpha, \beta, \mathbf{r})$. Therefore, for fixed f and $\mathbf{x}$, the label row vector $\boldsymbol{\ell} = \mathbf{L}\overline{\mathbf{x}}^{\mathrm{T}}$ is an affine-linear function of $(\alpha, \beta, \mathbf{r})$.
- **Special Piecewise Simulation:** The simulator $\mathsf{SimAKGS}(f, \mathbf{x}, \gamma) \to \widetilde{\boldsymbol{\ell}}$ takes as input an ABP $f : \mathbb{Z}_p^n \to \mathbb{Z}_p$, an attribute vector $\mathbf{x} \in \mathbb{Z}_p^n$, and a target value $\gamma \in \mathbb{Z}_p$, and outputs a label row vector $\widetilde{\boldsymbol{\ell}} \in \mathbb{Z}_p^m$. For every α, β and $\mathbf{x}$, if $\gamma = \alpha f(\mathbf{x}) + \beta$, then

  $$\left(\mathbf{L}_1\overline{\mathbf{x}}^{\mathrm{T}}, \ldots, \mathbf{L}_m\overline{\mathbf{x}}^{\mathrm{T}}\right) \approx_s \mathsf{SimAKGS}(f, \mathbf{x}, \gamma),$$

  where $(L_1, \ldots, L_m) \leftarrow \mathsf{Garble}(f, \alpha, \beta)$.

We now recall the AKGS instantiation for ABPs from [LL20a]. Let $f : \mathbb{Z}_p^n \to \mathbb{Z}_p$ be an ABP of size m. The standard ABP-to-determinant transformation via the adjacency matrix representation of the underlying graph structure gives an $(m-1)\times(m-1)$ matrix $\mathbf{M}_f(\mathbf{x})$ whose entries are affine functions of $\mathbf{x}$ and whose determinant is $f(\mathbf{x})$. Given $\alpha, \beta \in \mathbb{Z}_p$, one augments it to an $m \times m$ affine matrix

$$\mathbf{M}'_{f,\alpha,\beta}(\mathbf{x}) = \begin{pmatrix} \mathbf{M}_f(\mathbf{x}) & \mathbf{m}_{1,\beta}(\mathbf{x}) \\ \mathbf{m}_2^{\mathrm{T}} & \alpha \end{pmatrix}$$

whose determinant is $\alpha f(\mathbf{x}) + \beta$. Here $\mathbf{m}_{1,\beta}(\mathbf{x}) \in \mathbb{Z}_p^{m-1}$ and $\mathbf{m}_2 \in \mathbb{Z}_p^{m-1}$ are the standard augmentation vectors; the former is affine in $(\mathbf{x}, \beta)$ and the latter is fixed by the ABP shape. The garbling algorithm samples a row vector $\mathbf{r} \leftarrow\!\!\$\ \mathbb{Z}_p^{m-1}$ and defines

$$\mathbf{N}(\mathbf{r}) = \begin{pmatrix} \mathbf{I}_{m-1} & \mathbf{r}^{\mathrm{T}} \\ \mathbf{0}_{m-1} & 1 \end{pmatrix} \in \mathbb{Z}_p^{m\times m}.$$

It computes

$$\mathbf{M}^{\mathrm{c}}_{f,\alpha,\beta,\mathbf{r}}(\mathbf{x}) = \mathbf{M}'_{f,\alpha,\beta}(\mathbf{x})\mathbf{N}(\mathbf{r}) = \begin{pmatrix} \mathbf{M}_f(\mathbf{x}) & \mathbf{M}_f(\mathbf{x})\mathbf{r}^{\mathrm{T}} + \mathbf{m}_{1,\beta}(\mathbf{x}) \\ \mathbf{m}_2^{\mathrm{T}} & \mathbf{m}_2^{\mathrm{T}}\mathbf{r}^{\mathrm{T}} + \alpha \end{pmatrix}.$$

The m entries of the last column are the label functions $L_1(\mathbf{x}), \ldots, L_m(\mathbf{x})$, and Garble returns their coefficient row vectors $\mathbf{L}_1, \ldots, \mathbf{L}_m$. On input $(f, \mathbf{x}, \ell_1, \ldots, \ell_m)$, the evaluation algorithm forms the matrix $\mathbf{M}^{\mathrm{c}}_{f,\boldsymbol{\ell}}(\mathbf{x})$ whose first $m-1$ columns are the fixed columns determined by $\mathbf{M}_f(\mathbf{x})$ and whose last column is $\boldsymbol{\ell}^{\mathrm{T}}$, and returns its determinant. Since the determinant is linear in the last column, evaluation is linear in $\boldsymbol{\ell}$. The special simulator follows by sampling all but one last-column entry and reverse sampling the remaining entry so that the determinant equals the prescribed target.

4.2.2 The One-Key Core

We next describe the one-key secret-key core. This is a proof primitive rather than the final public-key AB-sIPFE.

Let $\Pi^{\mathsf{fh}} = (\mathsf{Setup}^{\mathsf{fh}}, \mathsf{KeyGen}^{\mathsf{fh}}, \mathsf{Enc}^{\mathsf{fh}}, \mathsf{Dec}^{\mathsf{fh}})$ be a secret-key function-hiding IPFE over the pairing group. The one-key core uses two independent copies. The inner-product copy has coordinate set

$$\mathfrak{s}_{\mathsf{1k,ip}} \coloneqq \{\mathsf{pad}\} \cup \mathfrak{s}_{\mathsf{pub}} \cup \mathfrak{s}_{\mathsf{priv}},$$

and the attribute copy has coordinate set

$$\mathfrak{s}_{\mathsf{1k,att}} \coloneqq \{\mathsf{const}\} \cup \{\mathsf{coef}_i : i \in [n]\} \cup \{\mathsf{sim}_1, \mathsf{sim}_?\}.$$

The coordinates sim_1 and $\mathsf{sim}_?$ are set to zero by the honest algorithms and are used only in the security analysis.

Construction 4.4 (One-key AB-sIPFE core). The one-key core $\Pi_{\mathsf{1k}} = (\mathsf{Setup}_{\mathsf{1k}}, \mathsf{KeyGen}_{\mathsf{1k}}, \mathsf{Enc}_{\mathsf{1k}}, \mathsf{Dec}_{\mathsf{1k}})$ is defined as follows.

- $\mathsf{Setup}_{1k}(1^\lambda, \mathfrak{s}_{pub}, \mathfrak{s}_{priv}) \to \mathsf{msk}_{1k}$: On input the security parameter and the two payload slot sets, the algorithm runs

$$\mathsf{msk}_{ip}^{fh} \leftarrow \mathsf{Setup}^{fh}(1^\lambda, \mathfrak{s}_{1k,ip}), \qquad \mathsf{msk}_{att}^{fh} \leftarrow \mathsf{Setup}^{fh}(1^\lambda, \mathfrak{s}_{1k,att})$$

  with independent randomness, and outputs $\mathsf{msk}_{1k} = (\mathsf{msk}_{ip}^{fh}, \mathsf{msk}_{att}^{fh})$.

- $\mathsf{KeyGen}_{1k}(\mathsf{msk}_{1k}, \mathbf{x}, [\![\mathbf{v}]\!]_2) \to \mathsf{sk}_{1k}$: On input the master secret key, an attribute vector $\mathbf{x} \in \mathbb{Z}_p^n$, and a function vector $\mathbf{v} = (\mathbf{v}_{pub}, \mathbf{v}_{priv}) \in \mathbb{Z}_p^{\mathfrak{s}_{pub} \cup \mathfrak{s}_{priv}}$ encoded in $\mathbb{G}_2$, define

copy	vector	pad	$\mathfrak{s}_{pub}$	$\mathfrak{s}_{priv}$	dimension
ip	$\mathbf{V}_{ip}$	1	$\mathbf{v}_{pub}$	$\mathbf{v}_{priv}$	$1 + \lvert\mathfrak{s}_{pub}\rvert + \lvert\mathfrak{s}_{priv}\rvert$

  and

copy	vector	const	coef_i	sim_1	$\mathsf{sim}_?$	dimension
att	$\mathbf{V}_{att}$	1	$\mathbf{x}[i]$	0	0	$n + 3$.

  The algorithm computes

$$K_{ip} \leftarrow \mathsf{KeyGen}^{fh}(\mathsf{msk}_{ip}^{fh}, [\![\mathbf{V}_{ip}]\!]_2), \qquad K_{att} \leftarrow \mathsf{KeyGen}^{fh}(\mathsf{msk}_{att}^{fh}, [\![\mathbf{V}_{att}]\!]_2),$$

  and outputs $\mathsf{sk}_{1k} = (\mathbf{x}, K_{ip}, K_{att})$.

- $\mathsf{Enc}_{1k}(\mathsf{msk}_{1k}, \pi = (f, \mathsf{mode}), [\![\mathbf{u}]\!]_1) \to \mathsf{ct}_{1k}$: On input the master secret key, a ciphertext policy π, and a message vector $\mathbf{u} = (\mathbf{u}_{pub}, \mathbf{u}_{priv}) \in \mathbb{Z}_p^{\mathfrak{s}_{pub} \cup \mathfrak{s}_{priv}}$ encoded in $\mathbb{G}_1$, the algorithm samples $\mu, \eta \leftarrow\!\!\$\ \mathbb{Z}_p$ and sets

$$(\alpha, \beta) = \begin{cases} (\mu, 0), & \mathsf{mode} = \mathsf{nz}, \\ (\eta, \mu), & \mathsf{mode} = \mathsf{zero}. \end{cases}$$

  It computes $(L_1, \ldots, L_m) \leftarrow \mathsf{Garble}(f, \alpha, \beta)$ and forms

copy	vector	pad	$\mathfrak{s}_{pub}$	$\mathfrak{s}_{priv}$	dimension
ip	$\mathbf{U}_{ip}$	μ	$\mathbf{u}_{pub}$	$\mathbf{u}_{priv}$	$1 + \lvert\mathfrak{s}_{pub}\rvert + \lvert\mathfrak{s}_{priv}\rvert$

  and, for each $j \in [m]$,

copy	vector	const	coef_i	sim_1	$\mathsf{sim}_?$	dimension
att	$\mathbf{U}_j$	$\mathbf{L}_j[\mathsf{const}]$	$\mathbf{L}_j[\mathsf{coef}_i]$	0	0	$n + 3$.

  It computes

$$C_{ip} \leftarrow \mathsf{Enc}^{fh}(\mathsf{msk}_{ip}^{fh}, [\![\mathbf{U}_{ip}]\!]_1), \qquad C_j \leftarrow \mathsf{Enc}^{fh}(\mathsf{msk}_{att}^{fh}, [\![\mathbf{U}_j]\!]_1) \quad (j \in [m]),$$

  and returns $\mathsf{ct}_{1k} = (\pi, C_{ip}, C_1, \ldots, C_m)$.

- $\mathsf{Dec}_{1k}(\mathsf{sk}_{1k}, \mathsf{ct}_{1k}) \to [\![d]\!]_t \vee \perp$: On input a one-key secret key and a one-key ciphertext, the algorithm parses $\mathsf{sk}_{1k} = (\mathbf{x}, K_{ip}, K_{att})$ and $\mathsf{ct}_{1k} = (\pi = (f, \mathsf{mode}), C_{ip}, C_1, \ldots, C_m)$. If $\pi(\mathbf{x}) = 0$, it outputs $\perp$. Otherwise it computes

$$[\![d_{ip}]\!]_t \leftarrow \mathsf{Dec}^{fh}(K_{ip}, C_{ip}), \qquad [\![\ell_j]\!]_t \leftarrow \mathsf{Dec}^{fh}(K_{att}, C_j) \quad (j \in [m]),$$

  sets

$$[\![\Gamma]\!]_t \coloneqq \mathsf{Eval}^{exp}(f, \mathbf{x}, [\![\ell_1]\!]_t, \ldots, [\![\ell_m]\!]_t),$$

  and computes

$$[\![\kappa]\!]_t \coloneqq \begin{cases} f(\mathbf{x})^{-1}[\![\Gamma]\!]_t, & \mathsf{mode} = \mathsf{nz}, \\ [\![\Gamma]\!]_t, & \mathsf{mode} = \mathsf{zero}. \end{cases}$$

  It outputs $[\![d_{ip}]\!]_t - [\![\kappa]\!]_t$.

Proposition 4.5 (Correctness of the one-key core). *If Π^{fh} and the AKGS are correct, then the one-key core returns $[\![\mathbf{v}\mathbf{u}^T]\!]_t$ whenever $\pi(\mathbf{x}) = 1$.*

Proof. Let μ, η be the encryption randomness. By correctness of IPFE,

$$\mathsf{Dec}^{\mathsf{fh}}(K_{\mathsf{ip}}, C_{\mathsf{ip}}) = [\![\mu + \mathbf{v}_{\mathsf{pub}}\mathbf{u}_{\mathsf{pub}}^{\mathsf{T}} + \mathbf{v}_{\mathsf{priv}}\mathbf{u}_{\mathsf{priv}}^{\mathsf{T}}]\!]_{\mathsf{t}} = [\![\mu + \mathbf{v}\mathbf{u}^{\mathsf{T}}]\!]_{\mathsf{t}}.$$

Also, $\mathsf{Dec}^{\mathsf{fh}}(K_{\mathsf{att}}, C_j) = [\![\mathbf{L}_j\overline{\mathbf{x}}^{\mathsf{T}}]\!]_{\mathsf{t}} = [\![L_j(\mathbf{x})]\!]_{\mathsf{t}}$ for each j. Hence

$$\Gamma = \mathsf{Eval}(f, \mathbf{x}, L_1(\mathbf{x}), \ldots, L_m(\mathbf{x})) = \alpha f(\mathbf{x}) + \beta.$$

If $\mathsf{mode} = \mathsf{nz}$ and $\pi(\mathbf{x}) = 1$, then $f(\mathbf{x}) \neq 0$ and $\Gamma/f(\mathbf{x}) = \mu$. If $\mathsf{mode} = \mathsf{zero}$ and $\pi(\mathbf{x}) = 1$, then $f(\mathbf{x}) = 0$ and $\Gamma = \mu$. Thus the value subtracted by the decryption algorithm is $[\![\mu]\!]_{\mathsf{t}}$ in both cases, and the output is $[\![\mathbf{v}\mathbf{u}^{\mathsf{T}}]\!]_{\mathsf{t}}$. □

Lemma 4.6 (One-key security). *Assume that Π^{fh} is function hiding and that the AKGS satisfies Definition 4.3. Then the one-key core is IND secure in the following one-key sense. The adversary receives at most one left-right key query $(\mathbf{x}, [\![\mathbf{v}_0]\!]_2, [\![\mathbf{v}_1]\!]_2)$ and at most one left-right ciphertext query $(\pi, [\![\mathbf{u}_0]\!]_1, [\![\mathbf{u}_1]\!]_1)$. If either $\pi(\mathbf{x}) = 0$ or $\mathbf{v}_0\mathbf{u}_0^{\mathsf{T}} = \mathbf{v}_1\mathbf{u}_1^{\mathsf{T}}$, then the two bit experiments are indistinguishable.*

Proof. Write $z_b = \mathbf{v}_b\mathbf{u}_b^{\mathsf{T}}$ for $b \in \{0, 1\}$. By function hiding of the two IPFE copies, the real transcript is indistinguishable from one in which the only values retained from the IPFE encodings are the target values

$$D_b = \mu + z_b, \qquad \Gamma_b = \begin{cases} \mu f(\mathbf{x}), & \mathsf{mode} = \mathsf{nz}, \\ \eta f(\mathbf{x}) + \mu, & \mathsf{mode} = \mathsf{zero}, \end{cases}$$

and in which the label row vector $\widetilde{\boldsymbol{\ell}}$ is sampled as $\mathsf{SimAKGS}(f, \mathbf{x}, \Gamma_b)$. The transformation is the one-key AKGS/IPFE hybrid of [LL20a]: first function hiding hides all information beyond the relevant inner products, and then the piecewise simulator replaces the real label values by simulated labels with the same target value.

If $\pi(\mathbf{x}) = 1$, the admissibility condition gives $z_0 = z_1$. Using the same μ, η and the same simulator randomness in the two bit worlds makes (D_b, Γ_b) and the simulated labels identical. If $\pi(\mathbf{x}) = 0$ and $\mathsf{mode} = \mathsf{nz}$, then $f(\mathbf{x}) = 0$, so $\Gamma_b = 0$ and $D_b = \mu + z_b$ is uniform independently of b. If $\pi(\mathbf{x}) = 0$ and $\mathsf{mode} = \mathsf{zero}$, then $f(\mathbf{x}) \neq 0$ and the map

$$(\mu, \eta) \longmapsto (\mu + z_b, \eta f(\mathbf{x}) + \mu)$$

is a bijection on $\mathbb{Z}_p^2$. Thus (D_b, Γ_b) is uniform on $\mathbb{Z}_p^2$, independently of b. In all cases the final hybrid is independent of the challenge bit. □

4.2.3 Bootstrapping the One-Key Core to Public-Key AB-sIPFE

We now bootstrap the one-key core to a public-key scheme supporting many keys. Let

$$\Pi' = (\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{EncPK}', \mathsf{EncSK}', \mathsf{Dec}')$$

be the standard two-slot sIPFE of Definition 3.5. By the slotted-IPFE construction of [LL20a], using the standard pairing IPFE tools of [Wee20], Π' can be instantiated from the standard MDDH_k assumption in the two source groups.

The user-facing payload slots are exactly $\mathfrak{s}_{\mathsf{pub}}$ and $\mathfrak{s}_{\mathsf{priv}}$. The construction additionally uses fixed internal coordinates. The public AKGS block is

$$\mathfrak{s}_{\mathsf{ag}} \coloneqq \{\mathsf{pad}_t : t \in [k]\} \cup \{\mathsf{const}_t : t \in [k]\} \cup \{\mathsf{coef}_{t,i} : t \in [k],\ i \in [n]\},$$

so $|\mathfrak{s}_{\mathsf{ag}}| = k(n+2)$. The proof-only scratch block is

$$\mathfrak{s}_{\mathsf{1abe}} \coloneqq \{\mathsf{attconst}_t : t \in [k]\} \cup \{\mathsf{attcoef}_{t,i} : t \in [k],\ i \in [n]\} \cup \{\mathsf{sim}_1, \mathsf{sim}_?\},$$

so $|\mathfrak{s}_{\mathsf{1abe}}| = k(n+1) + 2$. All honest algorithms set the scratch coordinates to zero. Both sIPFE copies are set up over

$$\widehat{\mathfrak{s}}_{\mathsf{pub}} \coloneqq \mathfrak{s}_{\mathsf{pub}} \cup \mathfrak{s}_{\mathsf{ag}}, \qquad \widehat{\mathfrak{s}}_{\mathsf{priv}} \coloneqq \mathfrak{s}_{\mathsf{priv}} \cup \mathfrak{s}_{\mathsf{1abe}}. \tag{1}$$

The following table summarizes the slot structure.

block	coordinates	size	use
$\mathfrak{s}_{\mathsf{pub}}$	given by the AB-sIPFE syntax	$\lvert\mathfrak{s}_{\mathsf{pub}}\rvert$	public payload slots
$\mathfrak{s}_{\mathsf{priv}}$	given by the AB-sIPFE syntax	$\lvert\mathfrak{s}_{\mathsf{priv}}\rvert$	private payload slots
$\mathfrak{s}_{\mathsf{ag}}$	$\mathsf{pad}_t, \mathsf{const}_t, \mathsf{coef}_{t,i}$	$k(n+2)$	public AKGS aggregation
$\mathfrak{s}_{\mathsf{1abe}}$	$\mathsf{attconst}_t, \mathsf{attcoef}_{t,i}, \mathsf{sim}_1, \mathsf{sim}_?$	$k(n+1)+2$	one-key proof scratch

Construction 4.7 (Ciphertext-policy AB-sIPFE for ABPs). The scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{EncPK}, \mathsf{EncSK}, \mathsf{Dec})$ is defined as follows.

- $\mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{priv}}) \to (\mathsf{mpk}, \mathsf{msk})$: On input the security parameter and the two payload slot sets, the algorithm forms the extended slot sets in (1), runs

$$(\mathsf{mpk}'_{\mathsf{ip}}, \mathsf{msk}'_{\mathsf{ip}}) \leftarrow \mathsf{Setup}'(1^\lambda, \widehat{\mathfrak{s}}_{\mathsf{pub}}, \widehat{\mathfrak{s}}_{\mathsf{priv}})$$

  and independently

$$(\mathsf{mpk}'_{\mathsf{att}}, \mathsf{msk}'_{\mathsf{att}}) \leftarrow \mathsf{Setup}'(1^\lambda, \widehat{\mathfrak{s}}_{\mathsf{pub}}, \widehat{\mathfrak{s}}_{\mathsf{priv}}).$$

  It outputs

$$\mathsf{mpk} = (\mathsf{mpk}'_{\mathsf{ip}}, \mathsf{mpk}'_{\mathsf{att}}), \qquad \mathsf{msk} = (\mathsf{msk}'_{\mathsf{ip}}, \mathsf{msk}'_{\mathsf{att}}).$$

- $\mathsf{KeyGen}(\mathsf{msk}, \mathbf{x}, [\![\mathbf{v}]\!]_2) \to \mathsf{sk}$: On input the master secret key, an attribute vector $\mathbf{x} \in \mathbb{Z}_p^n$, and a function vector $\mathbf{v} = (\mathbf{v}_{\mathsf{pub}}, \mathbf{v}_{\mathsf{priv}}) \in \mathbb{Z}_p^{\mathfrak{s}_{\mathsf{pub}} \cup \mathfrak{s}_{\mathsf{priv}}}$ encoded in $\mathbb{G}_2$, the algorithm samples a row vector $\mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_p^k$ and defines $\mathbf{V}_{\mathsf{ip}}, \mathbf{V}_{\mathsf{att}} \in \mathbb{Z}_p^{\widehat{\mathfrak{s}}_{\mathsf{pub}} \cup \widehat{\mathfrak{s}}_{\mathsf{priv}}}$ as follows:

	pad_t	const_t	$\mathsf{coef}_{t,i}$	$\mathfrak{s}_{\mathsf{pub}}$	$\mathfrak{s}_{\mathsf{priv}}$	$\mathfrak{s}_{\mathsf{1abe}}$
$\mathbf{V}_{\mathsf{ip}}$	$\mathbf{s}[t]$	0	0	$\mathbf{v}_{\mathsf{pub}}$	$\mathbf{v}_{\mathsf{priv}}$	$\mathbf{0}$
$\mathbf{V}_{\mathsf{att}}$	0	$\mathbf{s}[t]$	$\mathbf{s}[t]\mathbf{x}[i]$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$

  for every $t \in [k]$ and $i \in [n]$. The algorithm runs

$$K_{\mathsf{ip}} \leftarrow \mathsf{KeyGen}'(\mathsf{msk}'_{\mathsf{ip}}, [\![\mathbf{V}_{\mathsf{ip}}]\!]_2), \qquad K_{\mathsf{att}} \leftarrow \mathsf{KeyGen}'(\mathsf{msk}'_{\mathsf{att}}, [\![\mathbf{V}_{\mathsf{att}}]\!]_2),$$

  and outputs $\mathsf{sk} = (\mathbf{x}, K_{\mathsf{ip}}, K_{\mathsf{att}})$.

- $\mathsf{EncPK}(\mathsf{mpk}, \pi = (f, \mathsf{mode}), [\![\mathbf{u}_{\mathsf{pub}}]\!]_1) \to \mathsf{ct}$: On input the master public key, a policy π and a public-slot message vector $\mathbf{u}_{\mathsf{pub}} \in \mathbb{Z}_p^{\mathfrak{s}_{\mathsf{pub}}}$ encoded in $\mathbb{G}_1$, the algorithm samples row vectors $\boldsymbol{\mu}, \boldsymbol{\eta} \leftarrow\!\!\$\ \mathbb{Z}_p^k$. For every $t \in [k]$, it sets

$$(\alpha_t, \beta_t) = \begin{cases} (\boldsymbol{\mu}[t], 0), & \mathsf{mode} = \mathsf{nz}, \\ (\boldsymbol{\eta}[t], \boldsymbol{\mu}[t]), & \mathsf{mode} = \mathsf{zero}, \end{cases}$$

  and computes $(L_1^t, \ldots, L_m^t) \leftarrow \mathsf{Garble}(f, \alpha_t, \beta_t)$. It defines $\mathbf{U}_{\mathsf{ip}}, \mathbf{U}_1, \ldots, \mathbf{U}_m \in \mathbb{Z}_p^{\widehat{\mathfrak{s}}_{\mathsf{pub}} \cup \widehat{\mathfrak{s}}_{\mathsf{priv}}}$ by

	pad_t	const_t	$\mathsf{coef}_{t,i}$	$\mathfrak{s}_{\mathsf{pub}}$	$\mathfrak{s}_{\mathsf{priv}}$	$\mathfrak{s}_{\mathsf{1abe}}$
$\mathbf{U}_{\mathsf{ip}}$	$\boldsymbol{\mu}[t]$	0	0	$\mathbf{u}_{\mathsf{pub}}$	$\mathbf{0}$	$\mathbf{0}$
$\mathbf{U}_j$	0	$\mathbf{L}_j^t[\mathsf{const}]$	$\mathbf{L}_j^t[\mathsf{coef}_i]$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$

  for every $t \in [k]$, $i \in [n]$ and $j \in [m]$. Finally, it computes

$$C_{\mathsf{ip}} \leftarrow \mathsf{EncPK}'(\mathsf{mpk}'_{\mathsf{ip}}, [\![\mathbf{U}_{\mathsf{ip}}|_{\widehat{\mathfrak{s}}_{\mathsf{pub}}}]\!]_1)$$
$$C_j \leftarrow \mathsf{EncPK}'(\mathsf{mpk}'_{\mathsf{att}}, [\![\mathbf{U}_j|_{\widehat{\mathfrak{s}}_{\mathsf{pub}}}]\!]_1) \quad (j \in [m])$$

  and outputs $\mathsf{ct} = (\pi, C_{\mathsf{ip}}, C_1, \ldots, C_m)$.

- $\mathsf{EncSK}(\mathsf{msk}, \pi = (f, \mathsf{mode}), [\![\mathbf{u}]\!]_1) \to \mathsf{ct}$: On input the master secret key, a policy π and a full message vector $\mathbf{u} = (\mathbf{u}_{\mathsf{pub}}, \mathbf{u}_{\mathsf{priv}}) \in \mathbb{Z}_p^{\mathfrak{s}_{\mathsf{pub}} \cup \mathfrak{s}_{\mathsf{priv}}}$ encoded in $\mathbb{G}_1$, the algorithm samples $\boldsymbol{\mu}, \boldsymbol{\eta}$ and garbles f exactly as in

EncPK. It defines the same vectors as above, except that $\mathbf{U}_{\mathsf{ip}}|_{\mathsf{s}_{\mathsf{priv}}} = \mathbf{u}_{\mathsf{priv}}$. Thus the only nonzero private coordinates in honest secret-key encryption are the payload coordinates of $\mathbf{U}_{\mathsf{ip}}$. It computes

$$C_{\mathsf{ip}} \leftarrow \mathsf{EncSK}'(\mathsf{msk}'_{\mathsf{ip}}, [\![\mathbf{U}_{\mathsf{ip}}]\!]_1)$$
$$C_j \leftarrow \mathsf{EncSK}'(\mathsf{msk}'_{\mathsf{att}}, [\![\mathbf{U}_j]\!]_1) \quad (j \in [m]),$$

where every private coordinate of each $\mathbf{U}_j$ is zero. The algorithm outputs $\mathsf{ct} = (\pi, C_{\mathsf{ip}}, C_1, \ldots, C_m)$. By mode indistinguishability of Π', using EncPK' instead for the label ciphertexts gives an identical distribution.

- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to [\![d]\!]_{\mathsf{t}} \vee \perp$: On input an AB-sIPFE secret key and an AB-sIPFE ciphertext, the algorithm parses $\mathsf{sk} = (\mathbf{x}, K_{\mathsf{ip}}, K_{\mathsf{att}})$ and $\mathsf{ct} = (\pi = (f, \mathsf{mode}), C_{\mathsf{ip}}, C_1, \ldots, C_m)$. If $\pi(\mathbf{x}) = 0$, output $\perp$. Otherwise compute

$$[\![d_{\mathsf{ip}}]\!]_{\mathsf{t}} \leftarrow \mathsf{Dec}'(K_{\mathsf{ip}}, C_{\mathsf{ip}}), \qquad [\![\ell_j]\!]_{\mathsf{t}} \leftarrow \mathsf{Dec}'(K_{\mathsf{att}}, C_j) \quad (j \in [m]).$$

Set

$$[\![\Gamma]\!]_{\mathsf{t}} \coloneqq \mathsf{Eval}^{\mathsf{exp}}(f, \mathbf{x}, [\![\ell_1]\!]_{\mathsf{t}}, \ldots, [\![\ell_m]\!]_{\mathsf{t}})$$

and

$$[\![\kappa]\!]_{\mathsf{t}} \coloneqq \begin{cases} f(\mathbf{x})^{-1}[\![\Gamma]\!]_{\mathsf{t}}, & \mathsf{mode} = \mathsf{nz}, \\ [\![\Gamma]\!]_{\mathsf{t}}, & \mathsf{mode} = \mathsf{zero}. \end{cases}$$

Output $[\![d_{\mathsf{ip}}]\!]_{\mathsf{t}} - [\![\kappa]\!]_{\mathsf{t}}$.

Proposition 4.8 (Correctness). *If Π' and the AKGS are correct, then Construction 4.7 is correct.*

Proof. Fix $\pi = (f, \mathsf{mode})$ and $\mathbf{x}$ with $\pi(\mathbf{x}) = 1$, and write $\mathbf{u} = (\mathbf{u}_{\mathsf{pub}}, \mathbf{u}_{\mathsf{priv}})$ and $\mathbf{v} = (\mathbf{v}_{\mathsf{pub}}, \mathbf{v}_{\mathsf{priv}})$. Let $\mathbf{s}$ be the key randomness and let $\boldsymbol{\mu}, \boldsymbol{\eta}$ be the ciphertext randomness. By correctness of the inner-product sIPFE copy,

$$[\![d_{\mathsf{ip}}]\!]_{\mathsf{t}} = [\![\mathbf{s}\boldsymbol{\mu}^{\mathsf{T}} + \mathbf{v}_{\mathsf{pub}}\mathbf{u}_{\mathsf{pub}}^{\mathsf{T}} + \mathbf{v}_{\mathsf{priv}}\mathbf{u}_{\mathsf{priv}}^{\mathsf{T}}]\!]_{\mathsf{t}} = [\![\mathbf{s}\boldsymbol{\mu}^{\mathsf{T}} + \mathbf{v}\mathbf{u}^{\mathsf{T}}]\!]_{\mathsf{t}}.$$

For each label ciphertext,

$$\ell_j = \sum_{t \in [k]} \mathbf{s}[t]\mathbf{L}_j^t\bar{\mathbf{x}}^{\mathsf{T}} = \sum_{t \in [k]} \mathbf{s}[t]L_j^t(\mathbf{x}).$$

By linearity of the AKGS, these are the labels of an aggregate garbling with secrets

$$(\alpha, \beta) = \begin{cases} (\mathbf{s}\boldsymbol{\mu}^{\mathsf{T}}, 0), & \mathsf{mode} = \mathsf{nz}, \\ (\mathbf{s}\boldsymbol{\eta}^{\mathsf{T}}, \mathbf{s}\boldsymbol{\mu}^{\mathsf{T}}), & \mathsf{mode} = \mathsf{zero}. \end{cases}$$

Thus $\Gamma = \alpha f(\mathbf{x}) + \beta$. If $\mathsf{mode} = \mathsf{nz}$, then $f(\mathbf{x}) \neq 0$ and $f(\mathbf{x})^{-1}\Gamma = \mathbf{s}\boldsymbol{\mu}^{\mathsf{T}}$. If $\mathsf{mode} = \mathsf{zero}$, then $f(\mathbf{x}) = 0$ and $\Gamma = \langle \mathbf{s}, \boldsymbol{\mu} \rangle$. In either case the value subtracted by Dec is exactly $[\![\langle \mathbf{s}, \boldsymbol{\mu} \rangle]\!]_{\mathsf{t}}$, leaving $[\![\mathbf{v}\mathbf{u}^{\mathsf{T}}]\!]_{\mathsf{t}}$. □

Proposition 4.9 (Mode indistinguishability). *If Π' is mode indistinguishable, then Construction 4.7 is mode indistinguishable.*

Proof. When EncSK is invoked on a vector whose private-slot part is zero, $\mathbf{U}_{\mathsf{ip}}$ has no nonzero private coordinates. The label vectors $\mathbf{U}_j$ always have zero private coordinates. Mode indistinguishability of each underlying sIPFE copy therefore identifies the distributions obtained by EncPK' and EncSK' component by component. □

4.2.4 Security Analysis

We prove adaptive IND-CPA security by a sequence of indistinguishability hybrids. The proof uses the function-hiding part of the underlying slotted IPFE security in two places. First, for authorized key/ciphertext pairs, the payload copy can be changed from the zero-bit vector to the one-bit vector because the AB-sIPFE admissibility condition fixes the corresponding inner product. Second, for unauthorized pairs, the attribute/AKGS copy is switched as in the ABE proof of [LL20a], and a standard MDDH row switch turns the pad paired with the active key into a fresh target-group pad. The tables below are proof normal forms obtained by function-hiding hybrids. They are not simulation-based security claims.

Proposition 4.10 (Adaptive IND-CPA security). *Instantiate Π' by the standard two-slot sIPFE of [LL20a]. Assume MDDH (Assumption 3.1), and view the resulting Π' as satisfying the function-hiding formulation of Definition 3.5. If the ABP AKGS satisfies Definition 4.3, then Construction 4.7 is adaptively IND-CPA secure for the IND-CPA experiment defined in Figure 5. More concretely, for every PPT adversary $\mathcal{A}$ making at most Q_K key queries and Q_E encryption queries, and whose ABP policies have at most $m_{\max}$ garbling labels, there exist PPT adversaries $\mathcal{B}_{\mathsf{fh}}$ and $\mathcal{B}_{\mathsf{mddh}}$ such that*

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\Pi,\mathcal{A}}(\lambda) \le \mathsf{poly}(Q_K, Q_E, m_{\max}) \cdot \mathsf{Adv}^{\mathsf{fh}}_{\Pi',\mathcal{B}_{\mathsf{fh}}}(\lambda) + \mathsf{poly}(Q_K, Q_E, m_{\max}) \cdot \mathsf{Adv}^{\mathsf{MDDH}_k}_{\mathcal{B}_{\mathsf{mddh}}}(\lambda) + \mathsf{negl}(\lambda).$$

Proof. Fix a challenge bit $b \in \{0,1\}$. Let the q-th key query be $(\mathbf{x}_q, [\![\mathbf{v}_{q,0}]\!]_2, [\![\mathbf{v}_{q,1}]\!]_2)$ and the r-th encryption query be $(\pi_r = (f_r, \mathsf{mode}_r), [\![\mathbf{u}_{r,0}]\!]_1, [\![\mathbf{u}_{r,1}]\!]_1)$. Write $\mathbf{v}_{q,b} = (\mathbf{v}_{q,b,\mathrm{pub}}, \mathbf{v}_{q,b,\mathrm{priv}})$ and $\mathbf{u}_{r,b} = (\mathbf{u}_{r,b,\mathrm{pub}}, \mathbf{u}_{r,b,\mathrm{priv}})$, and define

$$z^{(b)}_{q,r} \coloneqq \mathbf{v}_{q,b,\mathrm{pub}} \mathbf{u}^{\mathsf{T}}_{r,b,\mathrm{pub}} + \mathbf{v}_{q,b,\mathrm{priv}} \mathbf{u}^{\mathsf{T}}_{r,b,\mathrm{priv}}. \tag{2}$$

The admissibility condition gives

$$[\![\mathbf{v}_{q,0,\mathrm{pub}}]\!]_2 = [\![\mathbf{v}_{q,1,\mathrm{pub}}]\!]_2 \qquad \text{for every } q, \tag{3}$$

$$z^{(0)}_{q,r} = z^{(1)}_{q,r} \qquad \text{whenever } \pi_r(\mathbf{x}_q) = 1. \tag{4}$$

$\mathsf{Hyb}^{(b)}_0$. Let $\mathsf{Hyb}^{(b)}_0$ be the real IND-CPA experiment with challenge bit b, except that label ciphertexts are always written in secret-key mode with zero private coordinates. This is identically distributed to the real experiment by Proposition 4.9.

$\mathsf{Hyb}^{(b)}_i$. For $i \in [0; Q_K]$, hybrid $\mathsf{Hyb}^{(b)}_i$ is defined as follows. For each processed key $q \le i$ and each ciphertext query r, the interaction between the q-th key and the r-th ciphertext is in the target form

pair	payload target	AKGS target	label distribution
(q,r)	$D^{(b)}_{q,r} = \omega_{q,r} + z^{(b)}_{q,r}$	$\widehat{\Gamma}_{q,r} = \begin{cases} \omega_{q,r} f_r(\mathbf{x}_q), & \mathsf{mode}_r = \mathsf{nz}, \\ \psi_{q,r} f_r(\mathbf{x}_q) + \omega_{q,r}, & \mathsf{mode}_r = \mathsf{zero}, \end{cases}$	$\mathsf{SimAKGS}(f_r, \mathbf{x}_q, \widehat{\Gamma}_{q,r})$.

(5)

Here $(\omega_{q,r}, \psi_{q,r})_{r \in [Q_E]}$ are fresh and uniform for each processed key q. All interactions involving unprocessed keys $q > i$ are real with challenge bit b. Importantly, the target table still gives the correct authorized decryption value: if $\pi_r(\mathbf{x}_q) = 1$, then

$$D^{(b)}_{q,r} - \begin{cases} f_r(\mathbf{x}_q)^{-1} \widehat{\Gamma}_{q,r}, & \mathsf{mode}_r = \mathsf{nz}, \\ \widehat{\Gamma}_{q,r}, & \mathsf{mode}_r = \mathsf{zero} \end{cases} = z^{(b)}_{q,r}.$$

Thus authorized entries are not discarded. They are carried through the hybrids as fixed inner-product values, and by (4) those values are the same in the two challenge branches.

We now show that $\mathsf{Hyb}^{(b)}_{i-1} \approx_c \mathsf{Hyb}^{(b)}_i$ for every $i \in [Q_K]$. This is exactly Lemma 4.11 applied to the active key $q^\star = i$. After all key rows have been processed, the final hybrid is described entirely by the (5). We couple $\mathsf{Hyb}^{(0)}_{Q_K}$ and $\mathsf{Hyb}^{(1)}_{Q_K}$ entry by entry, using in both worlds the same AKGS simulator randomness for every pair (q,r). It then suffices to couple the uniform pads $(\omega_{q,r}, \psi_{q,r})$ so that the payload target $D_{q,r}$ and the AKGS target $\widehat{\Gamma}_{q,r}$ of (5) agree across the two worlds, which in turn equalizes the simulated label vectors. If $\pi_r(\mathbf{x}_q) = 1$, then $z^{(0)}_{q,r} = z^{(1)}_{q,r}$ by (4), and the same pads $(\omega_{q,r}, \psi_{q,r})$ make both targets identical. If $\pi_r(\mathbf{x}_q) = 0$ and $\mathsf{mode}_r = \mathsf{nz}$, then $f_r(\mathbf{x}_q) = 0$, so $\widehat{\Gamma}_{q,r} = 0$ in both worlds regardless of the pads. Setting $\omega^{(1)}_{q,r} = \omega^{(0)}_{q,r} + z^{(0)}_{q,r} - z^{(1)}_{q,r}$ equalizes $D_{q,r}$ and is a bijection of the uniform pad $\omega_{q,r}$. If $\pi_r(\mathbf{x}_q) = 0$ and $\mathsf{mode}_r = \mathsf{zero}$, then $f_r(\mathbf{x}_q) \neq 0$, and setting

$$\omega^{(1)}_{q,r} = \omega^{(0)}_{q,r} + z^{(0)}_{q,r} - z^{(1)}_{q,r}, \qquad \psi^{(1)}_{q,r} = \psi^{(0)}_{q,r} + \frac{z^{(1)}_{q,r} - z^{(0)}_{q,r}}{f_r(\mathbf{x}_q)}$$

equalizes both $D_{q,r}$ and $\widehat{\Gamma}_{q,r}$. Since $f_r(\mathbf{x}_q) \neq 0$, the map $(\omega^{(0)}_{q,r}, \psi^{(0)}_{q,r}) \mapsto (\omega^{(1)}_{q,r}, \psi^{(1)}_{q,r})$ is a bijection of $\mathbb{Z}_p^2$. Since the pads are uniform and independent across pairs, each of these inner-products are preserved across both worlds, so

the coupled target tables, and hence the full transcripts, are identically distributed, and

$$\mathsf{Hyb}^{(0)}_{Q_K} \equiv \mathsf{Hyb}^{(1)}_{Q_K}.$$

Combining the Q_K one-key transitions for both challenge bits with this final identity proves the theorem. □

Lemma 4.11. *Fix a challenge bit b and a key-query index $q^\star$. Let $\mathbf{s}_{q^\star}$ be the row sampled in the $q^\star$-th key, and let $\boldsymbol{\mu}_r, \boldsymbol{\eta}_r \in \mathbb{Z}_p^k$ be the seed row vectors sampled in the r-th ciphertext query. Define*

$$\kappa_{q^\star,r} = \langle \mathbf{s}_{q^\star}, \boldsymbol{\mu}_r \rangle, \qquad \tau_{q^\star,r} = \langle \mathbf{s}_{q^\star}, \boldsymbol{\eta}_r \rangle.$$

Let $z^{(b)}_{q^\star,r}$ be as in (2). There is a sequence of hybrids, using only the fixed slot sets of (1), that replaces all information involving the $q^\star$-th key and all ciphertext queries by the target form

$$D^{(b)}_{q^\star,r} = \omega_{q^\star,r} + z^{(b)}_{q^\star,r}, \tag{6}$$

$$\widehat{\Gamma}_{q^\star,r} = \begin{cases} \omega_{q^\star,r} f_r(\mathbf{x}_{q^\star}), & \mathsf{mode}_r = \mathsf{nz}, \\ \psi_{q^\star,r} f_r(\mathbf{x}_{q^\star}) + \omega_{q^\star,r}, & \mathsf{mode}_r = \mathsf{zero}, \end{cases} \tag{7}$$

where $(\omega_{q^\star,r}, \psi_{q^\star,r})_r$ are uniform and independent. The labels seen by the active key against the r-th ciphertext are sampled from $\mathsf{SimAKGS}(f_r, \mathbf{x}_{q^\star}, \widehat{\Gamma}_{q^\star,r})$. All interactions not involving the $q^\star$-th key remain unchanged. The loss is polynomial in the number of ciphertext queries and in the largest garbling size, times the advantages against the function-hiding security of the underlying sIPFE and standard MDDH.

Proof. We write the active-key row and the ciphertext rows explicitly. In the real hybrid, the active key contributes the following vectors to the two sIPFE copies:

	pad_t	const_t	$\mathsf{coef}_{t,i}$	$\mathfrak{s}_{\mathrm{pub}}$	$\mathfrak{s}_{\mathrm{priv}}$	$\mathfrak{s}_{\mathrm{1abe}}$
$\mathbf{V}^{(q^\star)}_{\mathrm{ip}}$	$\mathbf{s}_{q^\star}[t]$	0	0	$\mathbf{v}_{q^\star,b,\mathrm{pub}}$	$\mathbf{v}_{q^\star,b,\mathrm{priv}}$	$\mathbf{0}$
$\mathbf{V}^{(q^\star)}_{\mathrm{att}}$	0	$\mathbf{s}_{q^\star}[t]$	$\mathbf{s}_{q^\star}[t]\mathbf{x}_{q^\star}[i]$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$.

For the r-th ciphertext query and label j, the ciphertext vectors are

	pad_t	const_t	$\mathsf{coef}_{t,i}$	$\mathfrak{s}_{\mathrm{pub}}$	$\mathfrak{s}_{\mathrm{priv}}$	$\mathfrak{s}_{\mathrm{1abe}}$
$\mathbf{U}^{(r)}_{\mathrm{ip}}$	$\boldsymbol{\mu}_r[t]$	0	0	$\mathbf{u}_{r,b,\mathrm{pub}}$	$\mathbf{u}_{r,b,\mathrm{priv}}$	$\mathbf{0}$
$\mathbf{U}^{(r)}_{j}$	0	$\mathbf{L}^t_{r,j}[\mathsf{const}]$	$\mathbf{L}^t_{r,j}[\mathsf{coef}_i]$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$.

Thus the active-row target values in the real hybrid are

component	value	meaning
payload	$D^{\mathrm{real},(b)}_{q^\star,r} = \kappa_{q^\star,r} + z^{(b)}_{q^\star,r}$	$\langle \mathbf{V}^{(q^\star)}_{\mathrm{ip}}, \mathbf{U}^{(r)}_{\mathrm{ip}} \rangle$
label j	$\Lambda_{q^\star,r,j} = \sum_{t\in[k]} \mathbf{s}_{q^\star}[t]\mathbf{L}^t_{r,j}\bar{\mathbf{x}}^{\mathrm{T}}_{q^\star}$	$\langle \mathbf{V}^{(q^\star)}_{\mathrm{att}}, \mathbf{U}^{(r)}_{j} \rangle$.

(8)

By AKGS linearity, the evaluated aggregate label target is

$$\Gamma^{\mathrm{real}}_{q^\star,r} = \begin{cases} \kappa_{q^\star,r} f_r(\mathbf{x}_{q^\star}), & \mathsf{mode}_r = \mathsf{nz}, \\ \tau_{q^\star,r} f_r(\mathbf{x}_{q^\star}) + \kappa_{q^\star,r}, & \mathsf{mode}_r = \mathsf{zero}. \end{cases} \tag{9}$$

We now pass through four sub-hybrids.

Hybrid G_0. This is the incoming hybrid before switching the row $q^\star$, with the active-row vectors and target values shown in (8) and (9).

Hybrid G_1. Apply adaptive function hiding of the payload sIPFE copy to the single active key row against all ciphertext queries at once. The hybrid keeps the public coordinates $\mathbf{v}_{q^\star,b,\mathrm{pub}}$ and $\mathbf{u}_{r,b,\mathrm{pub}}$ fixed, and keeps the

entire table of payload inner products

$$D^{\mathrm{real},(b)}_{q^\star,r} = \kappa_{q^\star,r} + z^{(b)}_{q^\star,r} \qquad (r \in [Q_E])$$

fixed. Only hidden coordinates of the proof normal form are modified. This step is exactly where function hiding of the sIPFE payload block is used: the incoming layout and the normal form expose the same table of payload inner products and agree on all public-slot key coordinates, so they are admissible inputs to the function-hiding challenger and hence computationally indistinguishable. Crucially, the row switch never opens or randomizes the adversary-supplied value $z^{(b)}_{q^\star,r}$: it is carried symbolically inside each payload target. This is what renders the payload block transparent to the value of $z^{(b)}_{q^\star,r}$, and is precisely what lets the theorem couple the two challenge branches on an authorized pair $\pi_r(\mathbf{x}_{q^\star}) = 1$ through the admissibility identity $z^{(0)}_{q^\star,r} = z^{(1)}_{q^\star,r}$ of (4).

Hybrid G_2**.** Apply adaptive function hiding of the attribute sIPFE copy to expose only the active-row label inner products $\Lambda_{q^\star,r,j}$ from (8). The public coordinates $(\mathsf{const}_t, \mathsf{coef}_{t,i})$ of every key remain unchanged. The normal form only adds a private scratch duplicate for the active row, namely

	$\mathsf{attconst}_t$	$\mathsf{attcoef}_{t,i}$	sim_1	$\mathsf{sim}_?$
$q = q^\star$	$\mathbf{s}_{q^\star}[t]$	$\mathbf{s}_{q^\star}[t]\mathbf{x}_{q^\star}[i]$	$*$	$*$
$q \neq q^\star$	0	0	0	0.

The entries marked $*$ are the two scalar relay coordinates used in the usual pre/post orientation of the one-key proof. Thus the same ciphertext component $C^{(r)}_j$ can be paired with every key, but any correction placed in $\mathfrak{s}_{1\mathrm{abe}}$ has zero inner product with non-active keys. Equivalently, the active row sees the label target $\Lambda_{q^\star,r,j}$, whereas every row $q \neq q^\star$ sees exactly the same attribute-copy target as in G_1. The two relay coordinates are reused sequentially in the label-by-label sub-hybrids.

Hybrid G_3**.** Replace the active-row label values for every ciphertext r by

$$(\widetilde{\Lambda}_{q^\star,r,1}, \ldots, \widetilde{\Lambda}_{q^\star,r,m_r}) \leftarrow \mathsf{SimAKGS}(f_r, \mathbf{x}_{q^\star}, \Gamma^{\mathrm{real}}_{q^\star,r}),$$

where $\Gamma^{\mathrm{real}}_{q^\star,r}$ is defined in (9). By the special piecewise simulation property of Definition 4.3, this step is statistically indistinguishable. The replacement does not require a label ciphertext to carry different plaintext labels for different keys: it changes only the active-row function-hiding target table of the attribute sIPFE copy; all other rows are unchanged.

Hybrid G_4**.** Collect the seed vectors of all ciphertexts into the matrix

$$\mathbf{M}_{\mathrm{seed}} = \left(\boldsymbol{\mu}_1^{\mathrm{T}} \cdots \boldsymbol{\mu}_{Q_E}^{\mathrm{T}} \parallel \boldsymbol{\eta}_1^{\mathrm{T}} \cdots \boldsymbol{\eta}_{Q_E}^{\mathrm{T}}\right) \in \mathbb{Z}_p^{k \times 2Q_E}.$$

Equivalently, $\mathbf{M}^{\mathrm{T}}_{\mathrm{seed}} \in \mathbb{Z}_p^{2Q_E \times k}$ is the matrix whose rows are the seed vectors appearing in the ciphertexts, and the active row $\mathbf{s}^{\mathrm{T}}_{q^\star} \in \mathbb{Z}_p^{k \times 1}$ is the secret column in the row-switching reduction. In the concrete sIPFE proof this is embedded as a standard source-group instance $\mathsf{MDDH}^1_{k,2Q_E}$. The reduction never needs discrete logarithms of $\mathbf{s}_{q^\star}$, $\boldsymbol{\mu}_r$ or $\boldsymbol{\eta}_r$, and the target-group products are obtained from the encoded source-group objects by pairing. At this point the switch is a change in the active-row target table of the concrete sIPFE proof, not a change to the ciphertext seed vectors themselves. Therefore all non-active key rows remain exactly as in the incoming hybrid. The switch replaces

$$\left(\langle \mathbf{s}_{q^\star}, \boldsymbol{\mu}_r \rangle, \langle \mathbf{s}_{q^\star}, \boldsymbol{\eta}_r \rangle\right)_{r \in [Q_E]}$$

by independent uniform values $(\omega_{q^\star,r}, \psi_{q^\star,r})_{r \in [Q_E]}$. When $2Q_E > k$, this instance follows from MDDH_k by the amplification of [EHK+13]. When $2Q_E \leq k$, conditioning on the full row rank of $\mathbf{M}^{\mathrm{T}}_{\mathrm{seed}}$ makes $\mathbf{M}^{\mathrm{T}}_{\mathrm{seed}}\mathbf{s}^{\mathrm{T}}_{q^\star}$ exactly uniform over $\mathbb{Z}_p^{2Q_E}$, and the rank-deficiency probability is $O(1/p)$. Since the AKGS labels and the simulator output are affine in the seed values, the reduction post-processes the switched encodings by group operations and pairings to obtain the encoded target values. The active-row target table becomes exactly (6) and (7).

□

4.3 Construction of Predicate-Hiding Mixed AB-sIPFE

We provide a simple generic compiler that transforms any non-predicate-hiding AB-sIPFE scheme into a predicate-hiding mixed AB-sIPFE scheme. The construction is a slight adaptation of a known compiler that combines ABE and MFE to obtain private linear broadcast encryption or, more generally, message-carrying MFE [GKW18, Zha25].

Construction 4.12 (Predicate-hiding mixed AB-sIPFE). The construction uses the following building blocks:

- a pairing group $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, e, p)$,
- an MFE scheme $\Pi_1 = (\mathsf{Setup}_1, \mathsf{KeyGen}_1, \mathsf{EncPK}_1, \mathsf{EncSK}_1, \mathsf{Dec}_1)$ for a function class $\mathcal{F} : \mathcal{X} \to \{0,1\}$, and
- an AB-sIPFE scheme $\Pi_2 = (\mathsf{Setup}_2, \mathsf{KeyGen}_2, \mathsf{EncPK}_2, \mathsf{EncSK}_2, \mathsf{Dec}_2)$ based on $\mathbb{G}$ whose policy class captures the decryption function of Π_1.

The construction of the predicate-hiding mixed AB-sIPFE scheme Π for the function class $\mathcal{F}$ works as follows:

- $\mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathrm{pub}}, \mathfrak{s}_{\mathrm{priv}})$ runs

$$(\mathsf{mpk}_1, \mathsf{msk}_1) \leftarrow \mathsf{Setup}_1(1^\lambda)$$
$$(\mathsf{mpk}_2, \mathsf{msk}_2) \leftarrow \mathsf{Setup}_2(1^\lambda, \mathfrak{s}_{\mathrm{pub}}, \mathfrak{s}_{\mathrm{priv}})$$

  and outputs $\mathsf{mpk} \coloneqq (\mathsf{mpk}_1, \mathsf{mpk}_2)$ and $\mathsf{msk} \coloneqq (\mathsf{msk}_1, \mathsf{msk}_2)$.
- $\mathsf{KeyGen}(\mathsf{msk}, x, [\![\mathbf{v}]\!]_2)$ parses $\mathsf{msk} = (\mathsf{msk}_1, \mathsf{msk}_2)$, runs

$$\mathsf{sk}_1 \leftarrow \mathsf{KeyGen}_1(\mathsf{msk}_1, x)$$
$$\mathsf{sk}_2 \leftarrow \mathsf{KeyGen}_2(\mathsf{msk}_2, \mathsf{sk}_1, [\![\mathbf{v}]\!]_2)$$

  and outputs $\mathsf{sk} \coloneqq \mathsf{sk}_2$.
- $\mathsf{EncPK}(\mathsf{mpk}, f, [\![\mathbf{u}_{\mathrm{pub}}]\!]_1)$ aborts if $f \neq \mathbf{1}$, where $\mathbf{1}$ denotes the constant function that evaluates to 1 on every input. Otherwise, the algorithm parses $\mathsf{mpk} = (\mathsf{mpk}_1, \mathsf{mpk}_2)$, runs

$$\mathsf{ct}_1 \leftarrow \mathsf{EncPK}_1(\mathsf{mpk}_1)$$
$$\mathsf{ct}_2 \leftarrow \mathsf{EncPK}_2(\mathsf{mpk}_2, C[\mathsf{ct}_1], [\![\mathbf{u}_{\mathrm{pub}}]\!]_1)$$

  and outputs $\mathsf{ct} \coloneqq \mathsf{ct}_2$. Here, $C[\mathsf{ct}_1](\cdot)$ denotes the circuit $\mathsf{Dec}_1(\cdot, \mathsf{ct}_1)$, i.e., MFE decryption of the hardwired MFE ciphertext ct_1.
- $\mathsf{EncSK}(\mathsf{msk}, f, [\![\mathbf{u}]\!]_1)$ parses $\mathsf{msk} = (\mathsf{msk}_1, \mathsf{msk}_2)$, runs

$$\mathsf{ct}_1 \leftarrow \mathsf{EncSK}_1(\mathsf{msk}_1, f)$$
$$\mathsf{ct}_2 \leftarrow \mathsf{EncSK}_2(\mathsf{msk}_2, C[\mathsf{ct}_1], [\![\mathbf{u}]\!]_1)$$

  and outputs $\mathsf{ct} \coloneqq \mathsf{ct}_2$.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$ parses $\mathsf{sk} = \mathsf{sk}_2$ and $\mathsf{ct} = \mathsf{ct}_2$. Then it outputs

$$[\![d]\!]_t \coloneqq \mathsf{Dec}_2(\mathsf{sk}_2, \mathsf{ct}_2).$$

Proposition 4.13 (Correctness and security). *The AB-sIPFE scheme Π in Construction 4.12 satisfies the following properties:*

- *If Π_1 and Π_2 are correct, then so is Π.*
- *If Π_1 is q-bounded mode indistinguishable and Π_2 is mode indistinguishable, then Π is q-bounded mode indistinguishable.*
- *If Π_1 is q-bounded predicate hiding, then so is Π.*
- *If Π_2 is q-bounded message hiding, then so is Π.*

The proof of Proposition 4.13 is easily adapted from [GKW18]. Combining the AB-sIPFE from Section 4.2 with the MFE of [Zha25] yields the following theorem.

Theorem 4.14. *Assuming* MDDH_k *on pairings, there exists a predicate-hiding mixed AB-sIPFE supporting ABPs.*

5 Decentralized Multi-Authority Two-Mode Attribute-Based Encryption

5.1 Definition

We define our notion of multi-authority two-mode ABE.

Encryption policies. In our model, an encryption policy consists of two components $\pi = (f, \mathbb{A})$. The first component, $f\colon \mathcal{GID} \to \{0,1\}$, specifies which user identifiers are authorized to decrypt; we refer to f as an *identity predicate*. The second component, a monotone access structure $\mathbb{A}$ over a set A of attributes, specifies which combinations of attributes are authorized. While attributes are managed by multiple authorities as in standard multi-authority ABE, the user identifier space $\mathcal{GID}$ is managed by a single authority designated by the encryptor. A user with identifier $\mathsf{gid} \in \mathcal{GID}$ can decrypt a ciphertext encrypted under π if $f(\mathsf{gid}) = 1$ and the user's attributes satisfy $\mathbb{A}$. Looking ahead, the two policy components serve distinct purposes: the access structure $\mathbb{A}$ will be used to realize the tracing policy, while the identity predicate f enables known traitor tracing techniques that selectively disable decryption for chosen users and infer information from the resulting gaps in the success probability of a pirate decoder.

Encryption modes. Compared with traditional MA-ABE, our notion of MA-2ABE features two encryption modes. In the public-key mode, the encryption algorithm EncPK takes only public information, namely the public keys of all authorities, including both the authority managing $\mathcal{GID}$ and the authorities controlling the attributes in A. In the secret-key mode, the encryption algorithm EncSK additionally has access to the master secret key of the authority managing $\mathcal{GID}$, while continuing to use the public keys of the attribute authorities.

When the public-mode encryption is sufficiently expressive to incorporate arbitrary identity predicates f, the secret-mode encryption algorithm EncSK becomes redundant. Indeed, the master secret key of the authority managing $\mathcal{GID}$ can always contain the corresponding public key, allowing EncSK to be implemented using EncPK.[10] Looking ahead, such a powerful public-mode encryption algorithm is precisely what enables the construction of traitor tracing schemes with public traceability.

Restricting public-mode encryption. Unfortunately, constructions achieving this level of expressiveness for public encryption do not attain optimal asymptotic parameters. Applications to traitor tracing instantiate f with comparison predicates over a set of $N = |\mathcal{GID}|$ users. Even in the single-authority setting, parameters of existing constructions [BW06, BCFG17] grow with $\sqrt{N}$. To obtain parameter sizes logarithmic in N, we therefore consider a restricted form of public-mode encryption in which EncPK only supports the constant predicate $\mathbf{1}$, which evaluates to 1 on every input. In this setting, EncSK is no longer implied by EncPK, since only the former can encrypt with respect to nontrivial identity predicates.

This restricted variant bears similarities with primitives commonly used in constructions of TT schemes with secret traceability, such as private linear broadcast encryption (PLBE) and mixed functional encryption (MFE). These primitives likewise feature an additional secret-mode encryption algorithm that allows encrypting with respect to a nontrivial predicate f. In fact, the MA-PLBE notion considered in Section 2.2 can be viewed as a special case of MA-2ABE with restricted public-mode encryption. Motivated by these connections to (message-carrying) mixed FE [GKW18, Zha25] and in analogy to a similar property for AB-sIPFE (Definition 4.2), we call such MA-2ABE schemes *mixed*.

Definition. We now give the formal definition of our MA-2ABE primitive.

Definition 5.1 (MA-2ABE). A decentralized multi-authority two-mode attribute-based encryption (MA-2ABE) scheme for monotone access structures and a class $\mathcal{F}\colon \mathcal{GID} \to \{0,1\}$ of identity predicates consists of six procedures defined below. We denote by $\mathcal{M}$ the supported message space and by $\mathcal{ID}$[11] and $\mathcal{GID}$ the (disjoint)

[10] We nevertheless keep the additional EncSK algorithm to have a unified syntax for all our MA-2ABE constructions.

[11] When using MA-2ABE as a building block in the construction of MA-AB-TT, we further split $\mathcal{ID}$ into the universe of attribute authorities $\mathcal{AID}$ and the universe of tracing authorities $\mathcal{TID}$.

universes of authority identifiers and global user identifiers, respectively. Additionally, we assume that each authority controls only one attribute, and hence we can use the words "authority" and "attribute" interchangeably. This definition naturally generalizes to the situation in which each authority potentially controls an arbitrary (bounded or unbounded) number of attributes (see [LW11, RW15]).

- $\mathsf{GlobalSetup}(1^\lambda) \to \mathsf{gp}$: On input the security parameter 1^λ, it outputs the global public parameters gp for the system. We assume that gp includes the descriptions of $\mathcal{ID}$ and $\mathcal{GID}$.
- $\mathsf{AuthSetup}(\mathsf{gp}, \mathsf{id}) \to (\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}})$: On input the global parameters gp and an authority identifier $\mathsf{id} \in \mathcal{ID}$, it outputs the authority's public/master secret key pair $(\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}})$.
- $\mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}}, \mathsf{gid}) \to \mathsf{sk}_{\mathsf{id,gid}}$: On input the global parameters gp, the master secret key $\mathsf{msk}_{\mathsf{id}}$ of an authority $\mathsf{id} \in \mathcal{ID}$, and a user's global identifier $\mathsf{gid} \in \mathcal{GID}$, it outputs a secret key $\mathsf{sk}_{\mathsf{id,gid}}$ for the user.
- $\mathsf{EncPK}(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu) \to \mathsf{ct}$: On input the global parameters gp, the public keys $\mathsf{pk}_{\mathsf{id}^*}$ of an authority $\mathsf{id}^* \in \mathcal{ID}$, the public keys $\{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}$ for a set of authorities $A \subseteq (\mathcal{ID} \setminus \{\mathsf{id}^*\})$, a predicate $\pi = (f, \mathbb{A})$ consisting of an identity predicate $f \in \mathcal{F}$ and a monotone access structure $\mathbb{A} \subseteq 2^{\mathcal{ID}}$ on $A^* := A \cup \{\mathsf{id}^*\}$, and a message $\mu \in \mathcal{M}$, it outputs a ciphertext ct.
- $\mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu) \to \mathsf{ct}$: On input the global parameter gp, the master secret key $\mathsf{msk}_{\mathsf{id}^*}$ of an authority $\mathsf{id}^* \in \mathcal{ID}$, the public keys $\{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}$ for a set of attribute authorities $A \subseteq (\mathcal{ID} \setminus \{\mathsf{id}^*\})$, a predicate $\pi = (f, \mathbb{A})$ consisting of an identity predicate $f \in \mathcal{F}$ and a monotone access structure $\mathbb{A} \subseteq 2^{\mathcal{ID}}$ on A^*, and a message $\mu \in \mathcal{M}$, it outputs a ciphertext ct.
- $\mathsf{Dec}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{id,gid}}\}_{\mathsf{id}\in A}, \mathsf{ct}) \to \mu' \vee \perp$: On input the global parameters gp, a set of secret keys $\{\mathsf{sk}_{\mathsf{id,gid}}\}_{\mathsf{id}\in A}$ issued by authorities $\mathsf{id} \in A$, and a ciphertext ct, it outputs a message $\mu' \in \mathcal{M}$ or $\perp$ (a designated error symbol).

Moreover, an MA-2ABE scheme Π should satisfy the following properties.

- **Correctness:** We say that Π is correct if there exist negligible functions $\mathsf{negl}_{\mathsf{pub}}, \mathsf{negl}_{\mathsf{priv}}$ such that for every $\lambda \in \mathbb{N}$, every message $\mu \in \mathcal{M}$, every $\mathsf{gid} \in \mathcal{GID}$, every $\mathsf{id}^* \in \mathcal{ID}$, every $A \subseteq (\mathcal{ID} \setminus \{\mathsf{id}^*\})$, every monotone access structure $\mathbb{A} \subseteq 2^{\mathcal{ID}}$ on A^*, every $S \in \mathbb{A}$, and every $f \in \mathcal{F}$ such that $f(\mathsf{gid}) = 1$, it holds that

$$\Pr\left[\mu = \mathsf{Dec}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{id,gid}}\}_{\mathsf{id}\in S}, \mathsf{ct}) \;\middle|\; \begin{array}{l} \mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda) \\ \forall \mathsf{id} \in A^* \cup S,\ (\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}}) \leftarrow \mathsf{AuthSetup}(\mathsf{gp}, \mathsf{id}) \\ \forall \mathsf{id} \in S,\ \mathsf{sk}_{\mathsf{id,gid}} \leftarrow \mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}}, \mathsf{gid}) \\ \mathsf{ct} \leftarrow \mathsf{EncPK}(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi = (f, \mathbb{A}), \mu) \end{array}\right] \geq 1 - \mathsf{negl}_{\mathsf{pub}}(\lambda),$$

$$\Pr\left[\mu = \mathsf{Dec}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{id,gid}}\}_{\mathsf{id}\in S}, \mathsf{ct}) \;\middle|\; \begin{array}{l} \mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda) \\ \forall \mathsf{id} \in A^* \cup S,\ (\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}}) \leftarrow \mathsf{AuthSetup}(\mathsf{gp}, \mathsf{id}) \\ \forall \mathsf{id} \in S,\ \mathsf{sk}_{\mathsf{id,gid}} \leftarrow \mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}}, \mathsf{gid}) \\ \mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi = (f, \mathbb{A}), \mu) \end{array}\right] \geq 1 - \mathsf{negl}_{\mathsf{priv}}(\lambda).$$

- **Mode indistinguishability:** Let $q(\cdot)$ be any polynomial. We say that Π is q-bounded mode indistinguishable if for all stateful PPT adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$, such that for all $\lambda \in \mathbb{N}$,

$$\left|\Pr\left[\mathsf{ExpMode}^{q,1}_{\Pi,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{ExpMode}^{q,0}_{\Pi,\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda),$$

  where the experiment $\mathsf{ExpMode}^{q,b}_{\Pi,\mathcal{A}}$ is defined in Figure 6 and the adversary has access to the oracles $\mathcal{O} := \{\mathcal{O}\mathsf{AuthSetup}, \mathcal{O}\mathsf{KeyGen}, \mathcal{O}\mathsf{EncSK}\}$.
- **Predicate hiding:** Let $q(\cdot)$ be any polynomial. We say that Π is q-bounded predicate hiding if for all stateful PPT adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$, such that for all $\lambda \in \mathbb{N}$,

$$\left|\Pr\left[\mathsf{ExpPred}^{q,1}_{\Pi,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{ExpPred}^{q,0}_{\Pi,\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{ExpPred}^{q,b}_{\Pi,\mathcal{A}}$ is defined in Figure 6 and the adversary has access to the oracles $\mathcal{O} \coloneqq \{\mathcal{O}\mathsf{AuthSetup}, \mathcal{O}\mathsf{KeyGen}, \mathcal{O}\mathsf{EncSK}\}$.

- **Message hiding:** Let $q(\cdot)$ be any polynomial. We say that Π is q-bounded message-hiding if for all stateful PPT adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$, such that for all $\lambda \in \mathbb{N}$,

$$\left|\Pr\left[\mathsf{ExpMsg}^{q,1}_{\Pi,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{ExpMsg}^{q,0}_{\Pi,\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{ExpMsg}^{q,b}_{\Pi,\mathcal{A}}$ is defined in Figure 6, the adversary has access to the oracles $\mathcal{O} \coloneqq \{\mathcal{O}\mathsf{AuthSetup}, \mathcal{O}\mathsf{KeyGen}, \mathcal{O}\mathsf{EncSK}, \mathcal{O}\mathsf{Corrupt}\}$ and $C' \coloneqq A \setminus \mathcal{K}$.

Remark 5.2 (Security against static corruptions). We say that Π is secure against static corruptions if the adversary $\mathcal{A}$ in the experiments $\mathsf{ExpMode}^{q,b}_{\Pi,\mathcal{A}}$, $\mathsf{ExpPred}^{q,b}_{\Pi,\mathcal{A}}$ and $\mathsf{ExpMsg}^{q,b}_{\Pi,\mathcal{A}}$ can query the oracle $\mathcal{O}\mathsf{Corrupt}(\mathsf{id})$ only before making the first query $\mathcal{O}\mathsf{KeyGen}(\mathsf{id}, \cdot)$ and before receiving the challenge ciphertext ct.[12]

Similarly to the case of AB-sIPFE (Definition 4.2), we call an MA-2ABE *mixed* if its public-mode encryption is restricted to $f = \mathbf{1}$.

Definition 5.3 (Mixed MA-2ABE). An MA-2ABE scheme $\Pi = (\mathsf{GlobalSetup}, \mathsf{AuthSetup}, \mathsf{KeyGen}, \mathsf{EncPK}, \mathsf{EncSK}, \mathsf{Dec})$ for a function class $\mathcal{F}$ is called mixed if the public-mode encryption algorithm EncPK is defined only for the constant function $\mathbf{1} \in \mathcal{F}$. That is, $\mathsf{EncPK}(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi = (f, \mathbb{A}), \mu)$ is supported if and only if $f = \mathbf{1}$.

It will be useful to rely on MA-2ABE schemes with an additional property which requires that the distribution of ciphertexts generated as public-mode encryptions with respect to $f = \mathbf{1}$ do not depend on the choice of id^*.

Definition 5.4 (Obliviousness to id^*). We say that Π is oblivious to the identity predicate encrypting authority id^* if for every $\lambda \in \mathbb{N}$, every monotone access structure $\mathbb{A}$ on a set $A \subseteq \mathcal{ID}$, every $\mathsf{id}^*_0, \mathsf{id}^*_1 \in A$, every message $\mu \in \mathcal{M}$, every public parameters gp and every set of public keys $\{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}$, the following two distributions are identical:

$$\begin{aligned}&\left\{\mathsf{ct}_0 \;\middle|\; \mathsf{ct}_0 \leftarrow \mathsf{EncPK}\left(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*_0}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A\setminus\{\mathsf{id}^*_0\}}, \pi = (\mathbf{1}, \mathbb{A}), \mu\right)\right\}\\ \equiv &\left\{\mathsf{ct}_1 \;\middle|\; \mathsf{ct}_1 \leftarrow \mathsf{EncPK}\left(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*_1}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A\setminus\{\mathsf{id}^*_1\}}, \pi = (\mathbf{1}, \mathbb{A}), \mu\right)\right\}.\end{aligned}$$

Looking ahead, in Proposition 6.6 we show that obliviousness to id^*, along with mode indistinguishability and message hiding, implies a natural notion of IND-CPA security.

5.2 Construction with Indistinguishability-Based Security

In this section, we provide our MA-2ABE construction with q-bounded indistinguishability-based security defined in Definition 5.1. In Section 5.3, we lift our construction to the corresponding decoder-based security notions required for applications to traitor tracing.

5.2.1 Construction

Our MA-2ABE scheme makes generic use of an AB-sIPFE scheme. In particular, if this AB-sIPFE is mixed, then the resulting MA-2ABE is also mixed.

Construction 5.5 ((Mixed) MA-2ABE). The construction uses the following building blocks:

- a pairing group $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, e, p)$,
- a (mixed) AB-sIFPE scheme $\Pi' = (\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{EncPK}', \mathsf{EncSK}', \mathsf{Dec}')$ based on $\mathbb{G}$ for a function class $\mathcal{F}$ where $f\colon \mathcal{GID} \to \{0,1\}$ for each $f \in \mathcal{F}$, and
- a hash function $H\colon \mathcal{GID} \to \mathbb{G}_2^k$ modelled as a random oracle.

[12] Note that security with static corruptions implies adaptive corruptions via a standard complexity leveraging argument that incurs a security loss of 2^Q where Q denotes the number of queries to $\mathcal{O}\mathsf{AuthSetup}$.

$\mathsf{ExpMode}^{q,b}_{\Pi,\mathcal{A}}(1^\lambda)$

$f^* \leftarrow \mathcal{A}(1^\lambda)$
$\mathcal{K}, Q \coloneqq [\cdot];\ q' \coloneqq 0$
$\mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$
$(\mathsf{id}^*, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \mathbb{A}, \mu) \leftarrow \mathcal{A}^{O}(\mathsf{gp})$
assert $\mathsf{id}^* \notin A \wedge \mathcal{K}[\mathsf{id}^*] \neq \bot$
parse $(\mathsf{pk}_{\mathsf{id}^*}, \mathsf{msk}_{\mathsf{id}^*}) \coloneqq \mathcal{K}[\mathsf{id}^*]$
$\mathsf{ct}_0 \leftarrow \mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi = (f^*, \mathbb{A}), \mu)$
$\mathsf{ct}_1 \leftarrow \mathsf{EncPK}(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi = (f^*, \mathbb{A}), \mu)$
$b' \leftarrow \mathcal{A}^{O}(\mathsf{ct}_b)$
assert $q' \leq q$
return b'

$\mathsf{ExpPred}^{q,b}_{\Pi,\mathcal{A}}(1^\lambda)$

$(f_0, f_1) \leftarrow \mathcal{A}(1^\lambda)$
$\mathcal{K}, Q \coloneqq [\cdot];\ q' \coloneqq 0$
$\mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$
$(\mathsf{id}^*, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \mathbb{A}, \mu) \leftarrow \mathcal{A}^{O}(\mathsf{gp})$
assert $\mathsf{id}^* \notin A \wedge \mathcal{K}[\mathsf{id}^*] \neq \bot$
parse $(\mathsf{pk}_{\mathsf{id}^*}, \mathsf{msk}_{\mathsf{id}^*}) \coloneqq \mathcal{K}[\mathsf{id}^*]$
$\mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi = (f_b, \mathbb{A}), \mu)$
$b' \leftarrow \mathcal{A}^{O}(\mathsf{ct})$
assert $q' \leq q \wedge \left[\forall \mathsf{gid} \in Q[\mathsf{id}^*],\ f_0(\mathsf{gid}) = f_1(\mathsf{gid})\right]$
return b'

$O\mathsf{AuthSetup}(\mathsf{id})$

assert $\mathcal{K}[\mathsf{id}] = \bot$
$(\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}}) \leftarrow \mathsf{AuthSetup}(\mathsf{gp}, \mathsf{id})$
$\mathcal{K}[\mathsf{id}] \coloneqq (\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}})$
return $\mathsf{pk}_{\mathsf{id}}$

$\mathsf{ExpMsg}^{q,b}_{\Pi,\mathcal{A}}(1^\lambda)$

$f^* \leftarrow \mathcal{A}(1^\lambda)$
$\mathcal{I}, \mathcal{K}, Q \coloneqq [\cdot];\ C \coloneqq \emptyset;\ q' \coloneqq 0$
$\mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$
$(\mathsf{id}^*, A, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in C'}, \mathbb{A}, \mu_0, \mu_1) \leftarrow \mathcal{A}^{O}(\mathsf{gp})$
assert $\mathsf{id}^* \notin (A \cup C) \wedge \mathcal{K}[\mathsf{id}^*] \neq \bot$
$\forall \mathsf{id} \in (A \setminus C') \cup \{\mathsf{id}^*\},$ **parse** $(\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}}) \coloneqq \mathcal{K}[\mathsf{id}]$
$\mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi = (f^*, \mathbb{A}), \mu_b)$
$b' \leftarrow \mathcal{A}^{O}(\mathsf{ct})$
$\forall \mathsf{gid},\ \mathcal{I}[\mathsf{gid}] \coloneqq \{\mathsf{id} : \mathsf{gid} \in Q[\mathsf{id}]\} \cup C \cup C'$
assert $q' \leq q \wedge (C \cup C') \notin \mathbb{A} \wedge$
  $\left[\forall \mathsf{gid} \in \bigcup_{\mathsf{id}\in A\cup\{\mathsf{id}^*\}} Q[\mathsf{id}],\ \mathcal{I}[\mathsf{gid}] \notin \mathbb{A} \vee\right.$
  $\left.(f^*(\mathsf{gid}) = 0 \wedge \mathcal{I}[\mathsf{gid}] \setminus \{\mathsf{id}^*\} \notin \mathbb{A})\right]$
return b'

$O\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid})$

assert $\mathcal{K}[\mathsf{id}] \neq \bot$
if $Q[\mathsf{id}] = \bot$ **then** $Q[\mathsf{id}] \coloneqq \emptyset$
$Q[\mathsf{id}] \coloneqq Q[\mathsf{id}] \cup \{\mathsf{gid}\}$
parse $(\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}}) \coloneqq \mathcal{K}[\mathsf{id}]$
return $\mathsf{sk}_{\mathsf{id},\mathsf{gid}} \leftarrow \mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}}, \mathsf{gid})$

$O\mathsf{EncSK}(\mathsf{id}^*, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu)$

$q' \coloneqq q' + 1$
assert $\mathcal{K}[\mathsf{id}^*] \neq \bot$
parse $(\mathsf{pk}_{\mathsf{id}^*}, \mathsf{msk}_{\mathsf{id}^*}) \coloneqq \mathcal{K}[\mathsf{id}^*]$
$\mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu)$
return ct

$O\mathsf{Corrupt}(\mathsf{id})$

assert $\mathcal{K}[\mathsf{id}] \neq \bot$
$C \coloneqq C \cup \{\mathsf{id}\}$
return $\mathcal{K}[\mathsf{id}]$

Figure 6: Experiments $\mathsf{ExpMode}^{q,b}_{\Pi,\mathcal{A}}$, $\mathsf{ExpPred}^{q,b}_{\Pi,\mathcal{A}}$ and $\mathsf{ExpMsg}^{q,b}_{\Pi,\mathcal{A}}$ for Definition 5.1

The construction of the (mixed) MA-2ABE scheme Π for monotone access structures and the identity function class $\mathcal{F}$ works as follows:

- $\mathsf{GlobalSetup}(1^\lambda)$ outputs $\mathsf{gp} \coloneqq (\mathbb{G}, H)$.
- $\mathsf{AuthSetup}(\mathsf{gp}, \mathsf{id})$ outputs

$$(\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}}) \leftarrow \mathsf{Setup}'\big(1^\lambda, \mathfrak{s}_{\mathrm{pub}} = [k+1], \mathfrak{s}_{\mathrm{priv}} = [k+2; k+n+3]\big).$$

- $\mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}}, \mathsf{gid})$ computes $[\![\mathbf{r}_{\mathsf{gid}}]\!]_2 \coloneqq H(\mathsf{gid})$ and outputs

$$\mathsf{sk}_{\mathsf{id},\mathsf{gid}} \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}_{\mathsf{id}}, \mathsf{gid}, [\![\mathbf{v}]\!]_2 = [\![(1, \mathbf{r}_{\mathsf{gid}}, \mathbf{0}_{n+2})]\!]_2\big).$$

- $\mathsf{EncPK}(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi = (f, \mathbb{A}), [\![\mu]\!]_{\mathrm{t}})$ parses $\mathbb{A} = (\mathbf{M} \in \mathbb{Z}_p^{n\times\ell}, \rho\colon [\ell] \to \mathcal{ID})$ and outputs $\perp$ if $\{\rho(j)\}_{j\in[\ell]} \not\subseteq A^*$ where $A^* \coloneqq A \cup \{\mathsf{id}^*\}$. Otherwise, it samples $\sigma \leftarrow\!\!\$\ \mathbb{Z}_p$, $\mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_p^{n-1}$, $\mathbf{T} \leftarrow\!\!\$\ \mathbb{Z}_p^{k\times(n-1)}$ and sets $[\![z]\!]_{\mathrm{t}} \coloneqq [\![\mu - \sigma]\!]_{\mathrm{t}}$. For each $j \in [\ell]$, it computes

$$\sigma_j \coloneqq (\sigma, \mathbf{s}) \cdot \mathbf{M}_j \in \mathbb{Z}_p, \qquad \boldsymbol{\tau}_j^{\mathrm{T}} \coloneqq (\mathbf{0}_k^{\mathrm{T}}, \mathbf{T}) \cdot \mathbf{M}_j \in \mathbb{Z}_p^{k\times 1},$$

  where $\mathbf{M}_j$ denotes the j-th column of $\mathbf{M}$, and runs

$$\mathsf{ct}_j \leftarrow \begin{cases} \mathsf{EncPK}'\big(\mathsf{pk}_{\rho(j)}, \mathbf{1}, [\![\mathbf{u}_{j,\mathrm{pub}}]\!]_1 = [\![(\sigma_j, \boldsymbol{\tau}_j)]\!]_1\big) & \text{if } \rho(j) \in A \\ \mathsf{EncPK}'\big(\mathsf{pk}_{\mathsf{id}^*}, f, [\![\mathbf{u}_{j,\mathrm{pub}}]\!]_1 = [\![(\sigma_j, \boldsymbol{\tau}_j)]\!]_1\big) & \text{if } \rho(j) = \mathsf{id}^* \end{cases}$$

  Here, $\mathbf{1}$ denotes the constant function that evaluates to 1 on every input. Finally, the algorithm outputs $\mathsf{ct} \coloneqq ([\![z]\!]_{\mathrm{t}}, \{\mathsf{ct}_j\}_{j\in[\ell]}, \mathbf{M}, \rho)$.

  Note: If Π is mixed, then EncPK is only defined when $f = \mathbf{1}$, so it suffices to instantiate Π' with a mixed AB-sIPFE scheme.

- $\mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi = (f, \mathbb{A}), \mu)$ is the same as $\mathsf{EncPK}(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, [\![\mu]\!]_{\mathrm{t}})$ except that for $j \in [\ell]$ such that $\rho(j) = \mathsf{id}^*$, the algorithm computes

$$\mathsf{ct}_j \leftarrow \mathsf{EncSK}'\big(\mathsf{msk}_{\mathsf{id}^*}, f, [\![\mathbf{u}_j]\!]_1 = [\![(\sigma_j, \boldsymbol{\tau}_j, \mathbf{0}_{n+2})]\!]_1\big).$$

- $\mathsf{Dec}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{id},\mathsf{gid}}\}_{\mathsf{id}\in A}, \mathsf{ct})$ parses $\mathsf{ct} = ([\![z]\!]_{\mathrm{t}}, \{\mathsf{ct}_j\}_{j\in[\ell]}, \mathbf{M}, \rho)$. If A is not authorized by $(\mathbf{M}, \rho)$, it outputs $\perp$. Otherwise, let $J \coloneqq \rho^{-1}(A)$ be the indices of columns of $\mathbf{M}$ associated with attributes in the user's attribute set A. Write $J = \{j_1, \ldots, j_{|J|}\}$. Let $\mathbf{M}_J$ be the matrix formed by taking the subset of columns in $\mathbf{M}$ indexed by elements in J. Since A is authorized, there exists a coefficient vector $\mathbf{c} = (c_1, \ldots, c_{|J|})$ such that $\mathbf{M}_J \cdot \mathbf{c}^{\mathrm{T}} = \mathbf{e}_1^{\mathrm{T}}$. For each $j \in J$, the algorithm runs

$$[\![d_j]\!]_{\mathrm{t}} \leftarrow \mathsf{Dec}'\big(\mathsf{sk}_{\rho(j),\mathsf{gid}}, \mathsf{ct}_j\big).$$

  Then it computes $[\![\sigma]\!]_{\mathrm{t}} \coloneqq \sum_{i\in[|J|]} c_i \cdot [\![d_{j_i}]\!]_{\mathrm{t}}$ and outputs $[\![\mu]\!]_{\mathrm{t}} \coloneqq [\![\sigma]\!]_{\mathrm{t}} + [\![z]\!]_{\mathrm{t}}$.

Before giving the proofs of correctness and security, we state the following theorem obtained by instantiating Π' with the AB-sIPFE schemes from Theorems A.3 and 4.14.

Theorem 5.6. *Assuming MDDH_k on pairings, there exist a mixed MA-2ABE supporting ABPs and an MA-2ABE supporting comparison predicates in the random oracle model.*

The detailed security properties are given below in Proposition 5.8. We now show correctness.

Proposition 5.7 (Correctness). *If Π' is correct, then so is Π.*

Proof. We verify correctness of EncSK; correctness of EncPK can be checked similarly. To encrypt a message $[\![\mu]\!]_{\mathrm{t}}$ with respect to a function f and a monotone access structure $\mathbb{A} = (\mathbf{M}, \rho)$, the secret encryption algorithm samples $\sigma \leftarrow\!\!\$\ \mathbb{Z}_p$, $\mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_p^{n-1}$, $\mathbf{T} \leftarrow\!\!\$\ \mathbb{Z}_p^{k\times(n-1)}$ and sets $[\![z]\!]_{\mathrm{t}} \coloneqq [\![\mu - \sigma]\!]_{\mathrm{t}}$. For each $j \in [\ell]$, it computes

$$\sigma_j \coloneqq (\sigma, \mathbf{s}) \cdot \mathbf{M}_j \in \mathbb{Z}_p, \qquad \boldsymbol{\tau}_j^{\mathrm{T}} \coloneqq (\mathbf{0}_k^{\mathrm{T}}, \mathbf{T}) \cdot \mathbf{M}_j \in \mathbb{Z}_p^{k\times 1},$$

and runs

$$\mathsf{ct}_j \leftarrow \begin{cases} \mathsf{EncPK}'\big(\mathsf{pk}_{\rho(j)}, \mathbf{1}, [\![\mathbf{u}_{j,\mathrm{pub}}]\!]_1 = [\![(\sigma_j, \boldsymbol{\tau}_j)]\!]_1\big) & \text{if } \rho(j) \in A \\ \mathsf{EncSK}'\big(\mathsf{msk}_{\mathsf{id}^*}, f, [\![\mathbf{u}_j]\!]_1 = [\![(\sigma_j, \boldsymbol{\tau}_j, \mathbf{0}_{n+2})]\!]_1\big) & \text{if } \rho(j) = \mathsf{id}^*. \end{cases}$$

Finally, the algorithm outputs $\mathsf{ct} \coloneqq ([\![z]\!]_{\mathrm{t}}, \{\mathsf{ct}_j\}_{j\in[\ell]}, \mathbf{M}, \rho)$. To generate a secret key for a pair $(\mathsf{id}, \mathsf{gid})$, the key generation algorithm computes

$$\mathsf{sk}_{\mathsf{id},\mathsf{gid}} \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}_{\mathsf{id}}, \mathsf{gid}, [\![\mathbf{v}]\!]_2 = [\![(1, \mathbf{r}_{\mathsf{gid}}, \mathbf{0}_{n+2})]\!]_2\big).$$

Given secret keys $\{\mathsf{sk}_{\mathsf{id},\mathsf{gid}}\}_{\mathsf{id}\in S}$ for some gid such that $f(\mathsf{gid}) = 1$ and $S \in \mathbb{A}$, the decryption algorithm computes a coefficient vector $\mathbf{c} = (c_1, \ldots, c_{|J|})$ such that $\mathbf{M}_J \cdot \mathbf{c}^{\mathrm{T}} = \mathbf{e}_1^{\mathrm{T}}$. Here, $J \coloneqq \rho^{-1}(A)$ and we parse $J = \{j_1, \ldots, j_{|J|}\}$. For all $j \in J$, the algorithm runs

$$[\![d_j]\!]_{\mathrm{t}} \leftarrow \mathsf{Dec}'\big(\mathsf{sk}_{\rho(j),\mathsf{gid}}, \mathsf{ct}_j\big).$$

By the correctness of Π', we have that $d_j = \sigma_j + \mathbf{r}_{\mathsf{gid}}\boldsymbol{\tau}_j^{\mathrm{T}}$. Then the algorithm computes $[\![\sigma']\!]_{\mathrm{t}} \coloneqq \sum_{i\in[|J|]} c_i \cdot [\![d_{j_i}]\!]_{\mathrm{t}}$ and outputs $[\![\mu]\!]_{\mathrm{t}} \coloneqq [\![\sigma']\!]_{\mathrm{t}} + [\![z]\!]_{\mathrm{t}}$. To conclude the proof, it suffices to verify that $\sigma' = \sigma$. We observe that

$$\begin{aligned} \sigma' = \sum_{i\in[|J|]} c_i \cdot d_{j_i} &= \sum_{i\in[|J|]} c_i \cdot (\sigma_{j_i} + \mathbf{r}_{\mathsf{gid}}\boldsymbol{\tau}_{j_i}^{\mathrm{T}}) \\ &= \sum_{i\in[|J|]} c_i \cdot \big((\sigma, \mathbf{s}) \cdot \mathbf{M}_{j_i} + \mathbf{r}_{\mathsf{gid}} \cdot (\mathbf{0}_k^{\mathrm{T}}, \mathbf{T}) \cdot \mathbf{M}_{j_i}\big) \\ &= (\sigma, \mathbf{s} + \mathbf{r}_{\mathsf{gid}}\mathbf{T}) \cdot \mathbf{M}_J \cdot \mathbf{c}^{\mathrm{T}} = \sigma, \end{aligned}$$

where the last equality follows from the fact that $\mathbf{M}_J \cdot \mathbf{c}^{\mathrm{T}} = \mathbf{e}_1^{\mathrm{T}}$. □

5.2.2 Security Analysis

Proposition 5.8 (Security). *The MA-2ABE scheme* Π *in Construction 5.5 satisfies the following properties:*

- *If* Π' *is q-bounded mode indistinguishable, then so is* Π.
- *If* Π' *is q-bounded predicate hiding, then so is* Π.
- *If* Π' *is q-bounded mode indistinguishable and q-bounded message hiding and* MDDH_k *holds in* $\mathbb{G}_2$*, then* Π *is q-bounded message hiding in the random oracle model.*
- Π *is oblivious to* id^*.

Proof. The proofs of mode indistinguishability, predicate hiding and obliviousness to id^* are immediate. We focus on the proof of message hiding.

We start by introducing the following notations. Let $M_{\mathrm{cor}} = \mathrm{span}\{\mathbf{M}_j : j \in [\ell], \rho(j) \in C \cup C'\}$ be the vector space spanned by the columns of $\mathbf{M}$ associated with corrupt authorities, and let $M_{\mathrm{cor}}^{\perp}$ denote its orthogonal complement. Furthermore, we let $A_{\mathrm{cor}}^{\perp}$ be the affine subspace of $M_{\mathrm{cor}}^{\perp}$ containing all vectors whose first coordinate is 1. Note that by the admissibility of $\mathcal{A}$, we have $\mathbf{e}_1 \notin M_{\mathrm{cor}}$, so $A_{\mathrm{cor}}^{\perp}$ is nonempty. We also denote $M_{\mathrm{cor},0}^{\perp} \coloneqq M_{\mathrm{cor}}^{\perp} \cap \{(0, \mathbf{x}) : \mathbf{x} \in \mathbb{Z}_p^{n-1}\}$ and let $\mathbf{B}_0 \in \mathbb{Z}_p^{n\times d}$ be a basis of $M_{\mathrm{cor},0}^{\perp}$ where $d \coloneqq \dim M_{\mathrm{cor},0}^{\perp}$.

For $\mathsf{gid} \in \mathcal{GID}$, we recall that $\mathcal{I}[\mathsf{gid}] \coloneqq \{\mathsf{id} \in \mathcal{ID} : \mathsf{gid} \in Q[\mathsf{id}]\} \cup C \cup C'$ denotes the set of authorities $\mathsf{id} \in \mathcal{ID}$ such that $\mathcal{A}$ submits a query $O\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid})$ or knows the master secret key $\mathsf{msk}_{\mathsf{id}}$. We define $M_{\mathsf{gid}} \coloneqq \mathrm{span}\{\mathbf{M}_j : j \in [\ell], \rho(j) \in \mathcal{I}[\mathsf{gid}]\}$, its orthogonal complement $M_{\mathsf{gid}}^{\perp}$, and the affine subspace $A_{\mathsf{gid}}^{\perp}$ of $M_{\mathsf{gid}}^{\perp}$ which contains all vectors whose first coordinate is 1. As above, the adversary's admissibility guarantees that $A_{\mathsf{gid}}^{\perp}$ is nonempty since $\mathbf{e}_1 \notin M_{\mathsf{gid}}$. Note that $A_{\mathsf{gid}}^{\perp}$ is a subset of $A_{\mathrm{cor}}^{\perp}$.

In the security analysis, we model the hash function H as a random oracle. We let $\mathcal{A}$ be an efficient adversary. For ease of exposition, we assume that $\mathcal{A}$ has the following properties:

- Algorithm $\mathcal{A}$ does not query any oracle (including H) on the same input more than once.
- Algorithm $\mathcal{A}$ always queries H on an input $\mathsf{gid} \in \mathcal{GID}$ before submitting a key generation query of the form $O\mathsf{KeyGen}(\cdot, \mathsf{gid})$.

- Algorithm $\mathcal{A}$ does not submit a key generation query of the form $O\mathsf{KeyGen}(\mathsf{id}, \cdot)$ such that $\mathsf{id} \in C \cup C'$.
- Algorithm $\mathcal{A}$ does not query $O\mathsf{AuthSetup}$ and $O\mathsf{Corrupt}$ anymore after seeing the challenge ciphertext.

We observe that all four properties hold without loss of generality, i.e. any efficient algorithm $\mathcal{A}$ that does not satisfy these properties can be generically compiled into an algorithm that does. We let $\mathsf{gid}_1, \ldots, \mathsf{gid}_{Q_{\mathrm{ro}}}$ denote the random oracle queries in the order they are made by $\mathcal{A}$ and define $\mathcal{G} \coloneqq \{\mathsf{gid}_i\}_{i \in [Q_{\mathrm{ro}}]}$.

We now proceed to the sequence of hybrid games, where $b \in \{0, 1\}$ indicates the challenge bit.

- $\mathsf{Hyb}_0^{(b)}$: This is the IND-CPA game against static corruptions with challenge bit b. We recall it here for clarity. First, the challenger initializes empty dictionaries $\mathcal{I}, \mathcal{K}, \mathcal{Q} \coloneqq [\cdot]$ and an empty set $C \coloneqq \emptyset$. Then it invokes the adversary $\mathcal{A}(1^\lambda)$ to receive the challenge identity policy f and sends the global parameters $\mathsf{gp} = (\mathbb{G}, H)$ in return. The adversary has access to the following oracles:
  - $O\mathsf{AuthSetup}(\mathsf{id})$: On input $\mathsf{id} \in \mathcal{ID}$, the challenger runs
  $$(\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}}) \leftarrow \mathsf{Setup}'\big(1^\lambda, \mathfrak{s}_{\mathrm{pub}} = [k+1], \mathfrak{s}_{\mathrm{priv}} = [k+2; k+n+3]\big),$$
  sets $\mathcal{K}[\mathsf{id}] \coloneqq (\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}})$ and outputs $\mathsf{pk}_{\mathsf{id}}$.
  - $O\mathsf{Corrupt}(\mathsf{id})$: On input $\mathsf{id} \in \mathcal{ID}$, the challenger adds id to C and outputs $\mathcal{K}[\mathsf{id}]$.
  - $O\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid})$: On input $\mathsf{id} \in \mathcal{ID}$ and $\mathsf{gid} \in \mathcal{GID}$, the challenger adds gid to $\mathcal{Q}[\mathsf{id}]$, computes $[\![\mathbf{r}_{\mathsf{gid}}]\!]_2 \coloneqq H(\mathsf{gid})$ and outputs
  $$\mathsf{sk}_{\mathsf{id},\mathsf{gid}} \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}_{\mathsf{id}}, \mathsf{gid}, [\![\mathbf{v}]\!]_2 = [\![(1, \mathbf{r}_{\mathsf{gid}}, \mathbf{0}_{n+2})]\!]_2\big).$$
  - $O\mathsf{EncSK}(\mathsf{id}^*_\kappa, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id} \in A_\kappa}, \pi_\kappa = (f_\kappa, \mathbb{A}_\kappa), \mu_\kappa)$ for $\kappa \in [q]$: On input $\mathsf{id}^*_\kappa \in \mathcal{ID}$, a set of public keys $\{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id} \in A_\kappa}$, a policy $\pi_\kappa = (f_\kappa, \mathbb{A}_\kappa = (\mathbf{M}_\kappa, \rho_\kappa))$ and a message μ_κ, the challenger samples $\sigma_\kappa \leftarrow\!\!\$\ \mathbb{Z}_p$, $\mathbf{s}_\kappa \leftarrow\!\!\$\ \mathbb{Z}_p^{n-1}$, $\mathbf{T}_\kappa \leftarrow\!\!\$\ \mathbb{Z}_p^{k \times (n-1)}$ and sets $[\![z_\kappa]\!]_{\mathrm{t}} \coloneqq [\![\mu_\kappa - \sigma_\kappa]\!]_{\mathrm{t}}$. For each $j \in [\ell]$, it computes $\sigma_{\kappa,j} \coloneqq (\sigma_\kappa, \mathbf{s}_\kappa) \cdot \mathbf{M}_{\kappa,j}$, $\boldsymbol{\tau}_{\kappa,j}^{\mathrm{T}} \coloneqq (\mathbf{0}_k^{\mathrm{T}}, \mathbf{T}_\kappa) \cdot \mathbf{M}_{\kappa,j}$ and runs
  $$\mathsf{ct}_{\kappa,j} \leftarrow \begin{cases} \mathsf{EncPK}'\big(\mathsf{pk}_{\rho_\kappa(j)}, \mathbf{1}, [\![\mathbf{u}_{\kappa,j,\mathrm{pub}}]\!]_1 = [\![(\sigma_{\kappa,j}, \boldsymbol{\tau}_{\kappa,j})]\!]_1\big) & \text{if } \rho_\kappa(j) \in A_\kappa \\ \mathsf{EncSK}'\big(\mathsf{msk}_{\mathsf{id}^*_\kappa}, f_\kappa, [\![\mathbf{u}_{\kappa,j}]\!]_1 = [\![(\sigma_{\kappa,j}, \boldsymbol{\tau}_{\kappa,j}, \mathbf{0}_{n+2})]\!]_1\big) & \text{if } \rho_\kappa(j) = \mathsf{id}^*_\kappa. \end{cases}$$
  Then the challenger outputs the ciphertext $\mathsf{ct} \coloneqq ([\![z_\kappa]\!]_{\mathrm{t}}, \{\mathsf{ct}_{\kappa,j}\}_{j \in [\ell]}, \mathbf{M}_\kappa, \rho_\kappa)$.

  At some point, the adversary outputs a set $A \subseteq \mathcal{ID}$, a set of (adversarially generated) public keys $\{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id} \in C'}$ where $C' \coloneqq A \setminus \mathcal{K}$, an access structure $\mathbb{A} = (\mathbf{M}, \rho)$ on A, and two messages $[\![\mu_0]\!]_{\mathrm{t}}, [\![\mu_1]\!]_{\mathrm{t}} \in \mathbb{G}_{\mathrm{t}}$. The challenger retrieves the (honestly generated) keys $(\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}}) \coloneqq \mathcal{K}[\mathsf{id}]$ for all $\mathsf{id} \in (C \cup \mathcal{H}) \cap A$, where $\mathcal{H} \coloneqq \mathcal{K} \setminus C$, and computes the challenge ciphertext as follows. First, it samples $\sigma \leftarrow\!\!\$\ \mathbb{Z}_p$, $\mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_p^{n-1}$, $\mathbf{T} \leftarrow\!\!\$\ \mathbb{Z}_p^{k \times (n-1)}$ and sets $[\![z]\!]_{\mathrm{t}} \coloneqq [\![\mu_b - \sigma]\!]_{\mathrm{t}}$. For each $j \in [\ell]$, it computes $\sigma_j \coloneqq (\sigma, \mathbf{s}) \cdot \mathbf{M}_j$, $\boldsymbol{\tau}_j^{\mathrm{T}} \coloneqq (\mathbf{0}_k^{\mathrm{T}}, \mathbf{T}) \cdot \mathbf{M}_j$ and runs
  $$\mathsf{ct}_j \leftarrow \begin{cases} \mathsf{EncPK}'\big(\mathsf{pk}_{\rho(j)}, \mathbf{1}, [\![\mathbf{u}_{j,\mathrm{pub}}]\!]_1 = [\![(\sigma_j, \boldsymbol{\tau}_j)]\!]_1\big) & \text{if } \rho(j) \in A \\ \mathsf{EncSK}'\big(\mathsf{msk}_{\mathsf{id}^*}, f, [\![\mathbf{u}_j]\!]_1 = [\![(\sigma_j, \boldsymbol{\tau}_j, \mathbf{0}_{n+2})]\!]_1\big) & \text{if } \rho(j) = \mathsf{id}^*. \end{cases}$$
  Then the challenger outputs the challenge ciphertext $\mathsf{ct} \coloneqq ([\![z]\!]_{\mathrm{t}}, \{\mathsf{ct}_j\}_{j \in [\ell]}, \mathbf{M}, \rho)$. After a second query phase with access to $O\mathsf{KeyGen}$ and $O\mathsf{EncSK}$, the adversary outputs a bit b'. If $\mathcal{A}$ is admissible, i.e. there does not exist $\mathsf{gid} \in \mathcal{GID}$ such that $\mathcal{I}[\mathsf{gid}] \in \mathbb{A}$, then the output of the experiment is b'. Otherwise, the output is set to 0.
- $\mathsf{Hyb}_1^{(b)}$: This is the same as $\mathsf{Hyb}_0^{(b)}$ except that the shares $\{\sigma_j\}_{j \in [\ell]}$ of σ are generated as
  $$\sigma_j \coloneqq (\sigma, \mathbf{s}) \cdot \mathbf{M}_j + \sigma(0, \mathbf{s}') \cdot \mathbf{M}_j,$$
  where $\mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_p^{n-1}$ as in $\mathsf{Hyb}_0^{(b)}$ and $\mathbf{s}' \in \mathbb{Z}_p^{n-1}$ is any vector such that $(1, \mathbf{s}') \in A_{\mathrm{cor}}^\perp$.

  To argue $\mathsf{Hyb}_0^{(b)} \equiv \mathsf{Hyb}_1^{(b)}$, we observe that the distributions $\{\mathbf{s} : \mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_p^{n-1}\}$ and $\{\mathbf{s} + \sigma\mathbf{s}' : \mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_p^{n-1}\}$ are identical for every $\mathbf{s}' \in \mathbb{Z}_p^{n-1}$. The first distribution corresponds to $\mathsf{Hyb}_0^{(b)}$ and the second distribution

corresponds to $\mathsf{Hyb}_1^{(b)}$, followed by the same post-processing.

Note that in $\mathsf{Hyb}_1^{(b)}$ the ciphertext components ct_j associated with corrupt authorities do not depend on σ anymore. Indeed, for $j \in [\ell]$ such that $\rho(j) \in C \cup C'$, we have

$$\sigma_j = (\sigma, \mathbf{s}) \cdot \mathbf{M}_j + \sigma(0, \mathbf{s}') \cdot \mathbf{M}_j = (0, \mathbf{s}) \cdot \mathbf{M}_j + \sigma(1, \mathbf{s}') \cdot \mathbf{M}_j = (0, \mathbf{s}) \cdot \mathbf{M}_j,$$

where the final equality follows from the orthogonality property of $(1, \mathbf{s}')$.

- $\mathsf{Hyb}_2^{(b)}$: This is the same as $\mathsf{Hyb}_1^{(b)}$ except that we modify the generation of the shares $\{\boldsymbol{\tau}_j\}_{j\in[\ell]}$. In this game, the challenger additionally samples $(1, \mathbf{t}'_1), \ldots, (1, \mathbf{t}'_k) \leftarrow\!\$\ A_{\mathsf{cor}}^{\perp}$ and sets

$$\mathbf{T}' \coloneqq \begin{pmatrix} — & \mathbf{t}'_1 & — \\ & \vdots & \\ — & \mathbf{t}'_k & — \end{pmatrix} \qquad \text{and} \qquad \boldsymbol{\tau}_j^{\mathrm{T}} \coloneqq (\mathbf{0}_k^{\mathrm{T}}, \mathbf{T}) \cdot \mathbf{M}_j + \boxed{(\mathbf{0}_k^{\mathrm{T}}, \mathbf{T}') \cdot \mathbf{M}_j} .$$

We have $\mathsf{Hyb}_1^{(b)} \equiv \mathsf{Hyb}_2^{(b)}$ by a similar argument as $\mathsf{Hyb}_0^{(b)} \equiv \mathsf{Hyb}_1^{(b)}$. First, the distributions $\{\mathbf{T} : \mathbf{T} \leftarrow\!\$\ \mathbb{Z}_p^{k\times(n-1)}\}$ and $\{\mathbf{T} + \mathbf{T}' : \mathbf{T} \leftarrow\!\$\ \mathbb{Z}_p^{k\times(n-1)}\}$ are identical for every $\mathbf{T}' \in \mathbb{Z}_p^{k\times(n-1)}$. Then we observe that the first distribution corresponds to $\mathsf{Hyb}_1^{(b)}$ and the second distribution corresponds to $\mathsf{Hyb}_2^{(b)}$, followed by the same post-processing.

- $\mathsf{Hyb}_3^{(b)}$: This is the same as $\mathsf{Hyb}_2^{(b)}$ except that we modify the sampling of $\mathbf{T}'$. In this game, the challenger picks $\mathbf{a} \in A_{\mathsf{cor}}^{\perp}$, samples $\mathbf{Y} \leftarrow\!\$\ \mathbb{Z}_p^{k\times d}$ and sets $(\mathbf{1}_k^{\mathrm{T}}, \mathbf{T}') \coloneqq \boxed{(\mathbf{1}_k^{\mathrm{T}} \otimes \mathbf{a}) + \mathbf{Y} \cdot \mathbf{B}_0^{\mathrm{T}}}$.

  We have $\mathsf{Hyb}_2^{(b)} \equiv \mathsf{Hyb}_3^{(b)}$ which follows from the fact that $\mathbf{A}_{\mathsf{cor}}^{\perp} = \mathbf{a} + \mathbf{M}_{\mathsf{cor},0}^{\perp}$ and the observation that $\mathbf{y} \cdot \mathbf{B}_0^{\mathrm{T}}$ for $\mathbf{y} \leftarrow\!\$\ \mathbb{Z}_p^d$ is distributed uniformly on $\mathbf{M}_{\mathsf{cor},0}^{\perp}$.

- $\mathsf{Hyb}_4^{(b)}$: This is the same as $\mathsf{Hyb}_3^{(b)}$ except that we change the computation of the challenge ciphertext ct. For all $j \in [\ell]$ such that $\rho(j) \in \mathcal{H} \setminus \{\mathsf{id}^*\}$, the challenger computes

$$\mathsf{ct}_j \leftarrow \boxed{\mathsf{EncSK}'\big(\mathsf{msk}_{\rho(j)}, \mathbf{1}, [\![\mathbf{u}_j]\!]_1 = [\![(\sigma_j, \boldsymbol{\tau}_j, \mathbf{0}_{n+2})]\!]_1\big)} ,$$

  where $\mathbf{1}$ denotes the constant function that evaluates to 1 on every input. We have $\mathsf{Hyb}_3^{(b)} \equiv \mathsf{Hyb}_4^{(b)}$ by the q-bounded mode indistinguishability of Π' ($\mathsf{ExpMode}^{q,b}_{\Pi',\mathcal{A}'}$ in Figure 4). In particular, note that the function f^* is known upfront, so $\mathsf{ExpMode}^{q,b}_{\Pi',\mathcal{A}'}$ can be invoked correctly.

- $\mathsf{Hyb}_5^{(b)}$: This is the same as $\mathsf{Hyb}_4^{(b)}$ except that the challenger samples the encryption randomness $(\sigma, \mathbf{s}, \mathbf{s}', \mathbf{T}, \mathbf{T}')$ at the outset of the game and embeds it in the secret keys. Specifically, upon receiving a key generation query $O\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid})$, the challenger outputs

$$\mathsf{sk}_{\mathsf{id},\mathsf{gid}} \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}_{\mathsf{id}}, \mathsf{gid}, [\![\mathbf{v}]\!]_2 = [\![(1, \mathbf{r}_{\mathsf{gid}}, \boxed{\mathbf{v}'_{\mathsf{gid}}}, \mathbf{0}_2)]\!]_2\big),$$

  where $\mathbf{v}'_{\mathsf{gid}} = (\sigma, \mathbf{s} + \sigma\mathbf{s}') + \mathbf{r}_{\mathsf{gid}}(\mathbf{0}_k^{\mathrm{T}}, \mathbf{T} + \mathbf{T}')$. Correspondingly, the challenge ciphertext components ct_j such that $\rho(j) \in \mathcal{H}$ are computed as

$$\mathsf{ct}_j \leftarrow \begin{cases} \mathsf{EncSK}'\big(\mathsf{msk}_{\rho(j)}, \mathbf{1}, [\![\mathbf{u}_j]\!]_1 = [\![(\ \boxed{\mathbf{0}_{1+k}, \mathbf{M}_j^{\mathrm{T}}}, \mathbf{0}_2)]\!]_1\big) & \text{if } \rho(j) \in A \\ \mathsf{EncSK}'\big(\mathsf{msk}_{\mathsf{id}^*}, f, [\![\mathbf{u}_j]\!]_1 = [\![(\ \boxed{\mathbf{0}_{1+k}, \mathbf{M}_j^{\mathrm{T}}}, \mathbf{0}_2)]\!]_1\big) & \text{if } \rho(j) = \mathsf{id}^*. \end{cases}$$

  We observe that the inner products between message and function vectors do not change from $\mathsf{Hyb}_4^{(b)}$ to $\mathsf{Hyb}_5^{(b)}$

$$\begin{aligned} (1, \mathbf{r}_{\mathsf{gid}}, \mathbf{0}_{n+2})(\sigma_j, \boldsymbol{\tau}_j, \mathbf{0}_{n+2})^{\mathrm{T}} &= \big(1, \mathbf{r}_{\mathsf{gid}}, \mathbf{0}_{n+2}\big) \cdot \big((\sigma, \mathbf{s} + \sigma\mathbf{s}') \cdot \mathbf{M}_j, ((\mathbf{0}_k^{\mathrm{T}}, \mathbf{T} + \mathbf{T}') \cdot \mathbf{M}_j)^{\mathrm{T}}, \mathbf{0}_{n+2}\big)^{\mathrm{T}} \\ &= \big(1, \mathbf{r}_{\mathsf{gid}}, (\sigma, \mathbf{s} + \sigma\mathbf{s}') + \mathbf{r}_{\mathsf{gid}}(\mathbf{0}_k^{\mathrm{T}}, \mathbf{T} + \mathbf{T}'), \mathbf{0}_2\big) \cdot \big(\mathbf{0}_{1+k}, \mathbf{M}_j^{\mathrm{T}}, \mathbf{0}_2\big)^{\mathrm{T}} \\ &= \big(1, \mathbf{r}_{\mathsf{gid}}, \mathbf{v}'_{\mathsf{gid}}, \mathbf{0}_2\big) \cdot \big(\mathbf{0}_{1+k}, \mathbf{M}_j^{\mathrm{T}}, \mathbf{0}_2\big)^{\mathrm{T}}, \end{aligned}$$

where the first inner product corresponds to $\mathsf{Hyb}_4^{(b)}$ and the last inner product corresponds to $\mathsf{Hyb}_5^{(b)}$. Then it follows from the q-bounded message hiding property of Π' that $\mathsf{Hyb}_4^{(b)} \approx_c \mathsf{Hyb}_5^{(b)}$.

- $\mathsf{Hyb}_6^{(b)}$: This game is the same as $\mathsf{Hyb}_5^{(b)}$ except that in responses to key queries $\mathcal{O}\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid})$, the challenger sets

$$\mathbf{v}'_{\mathsf{gid}} \coloneqq (\sigma, \mathbf{s} + \sigma\mathbf{s}') + \mathbf{r}_{\mathsf{gid}}(\mathbf{0}_k^\mathsf{T}, \mathbf{T} + \mathbf{T}') + \mathbf{y}_{\mathsf{gid}} \cdot \mathbf{B}_0^\mathsf{T},$$

where $\mathbf{y}_{\mathsf{gid}} \leftarrow\!\!\$\ \mathbb{Z}_p^d$ is a random vector sampled for each user identifier $\mathsf{gid} \in \mathcal{G}$.[13]

We argue that $\mathsf{Hyb}_5^{(b)} \approx_c \mathsf{Hyb}_6^{(b)}$ under the MDDH_k assumption in $\mathbb{G}_2$. Given a challenge $([\![\mathbf{R}]\!]_2, [\![\mathbf{U}]\!]_2) \in \mathbb{G}_2^{Q_{\mathrm{ro}} \times k} \times \mathbb{G}_2^{Q_{\mathrm{ro}} \times d}$, parse

$$\mathbf{R} \coloneqq \begin{pmatrix} — & \mathbf{r}_{\mathsf{gid}_1} & — \\ & \vdots & \\ — & \mathbf{r}_{\mathsf{gid}_{Q_{\mathrm{ro}}}} & — \end{pmatrix} \quad \text{and} \quad \mathbf{U} \coloneqq \begin{pmatrix} — & \mathbf{u}_{\mathsf{gid}_1} & — \\ & \vdots & \\ — & \mathbf{u}_{\mathsf{gid}_{Q_{\mathrm{ro}}}} & — \end{pmatrix}.$$

During the query phase, the simulation behaves as follows:

  - Upon $\mathcal{A}$ submitting the i-th random oracle query gid_i for $i \in [Q_{\mathrm{ro}}]$, the challenger programs the random oracle $H(\mathsf{gid}_i) \coloneqq [\![\mathbf{r}_{\mathsf{gid}_i}]\!]_2$.
  - Upon $\mathcal{A}$ submitting a key generation query $\mathcal{O}\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid}_i)$ for $i \in [Q_{\mathrm{ro}}]$, the challenger sets

$$[\![\mathbf{v}'_{\mathsf{gid}}]\!]_2 \coloneqq [\![(\sigma, \mathbf{s} + \sigma\mathbf{s}') + \mathbf{r}_{\mathsf{gid}}(-\mathbf{1}_k^\mathsf{T}, \mathbf{T}) + \mathbf{r}_{\mathsf{gid}}(\mathbf{1}_k^\mathsf{T} \otimes \mathbf{a}) + \mathbf{u}_{\mathsf{gid}} \cdot \mathbf{B}_0^\mathsf{T}]\!]_2.$$

Then we can observe that the challenger simulates $\mathsf{Hyb}_5^{(b)}$ if $\mathbf{U} = \mathbf{RY}$ for $\mathbf{Y} \leftarrow\!\!\$\ \mathbb{Z}_p^{k \times d}$, and it simulates $\mathsf{Hyb}_6^{(b)}$ if $\mathbf{U} \leftarrow\!\!\$\ \mathbb{Z}_p^{Q_{\mathrm{ro}} \times d}$.

- $\mathsf{Hyb}_7^{(b)}$: This game is the same as $\mathsf{Hyb}_6^{(b)}$ except that in responses to key queries $\mathcal{O}\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid})$, the challenger sets

$$\mathbf{v}'_{\mathsf{gid}} \coloneqq (\sigma, \mathbf{s} + \sigma\mathbf{s}') + \mathbf{r}_{\mathsf{gid}}(\mathbf{0}_k^\mathsf{T}, \mathbf{T} + \mathbf{T}') + \sigma \cdot \mathbf{y}_{\mathsf{gid}} \cdot \mathbf{B}_0^\mathsf{T}$$

We have $\mathsf{Hyb}_6^{(b)} \approx_s \mathsf{Hyb}_7^{(b)}$ by a simple statistical argument. The distributions $(\sigma, \{\mathbf{y}_{\mathsf{gid}_i}\}_{i \in [Q_{\mathrm{ro}}]})$ and $(\sigma, \{\sigma\mathbf{y}_{\mathsf{gid}_i}\}_{i \in [Q_{\mathrm{ro}}]})$ have statistical distance $1/p$, where the probability is taken over the choice of $\sigma \leftarrow\!\!\$\ \mathbb{Z}_p$ and $\{\mathbf{y}_{\mathsf{gid}_i} \leftarrow\!\!\$\ \mathbb{Z}_p^d\}_{i \in [Q_{\mathrm{ro}}]}$. Then the indistinguishability follows by observing that the first distribution corresponds to $\mathsf{Hyb}_6^{(b)}$ and the second distribution corresponds to $\mathsf{Hyb}_7^{(b)}$, both followed by the same post-processing.

- $\mathsf{Hyb}_{8,i}^{(b)}$ for $i \in [0; Q_{\mathrm{ro}}]$: This game is the same as $\mathsf{Hyb}_7^{(b)}$ except that, upon $\mathcal{A}$ submitting a key generation query $\mathcal{O}\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid}_{i'})$ such that $i' \in [i]$, the challenger sets

$$\mathbf{v}'_{\mathsf{gid}_{i'}} \coloneqq (0, \mathbf{s}) + \mathbf{r}_{\mathsf{gid}_{i'}}(\mathbf{0}_k^\mathsf{T}, \mathbf{T} + \mathbf{T}') + \sigma\mathbf{y}_{\mathsf{gid}_{i'}} \cdot \mathbf{B}_0^\mathsf{T}.$$

We note that $\mathsf{Hyb}_7^{(b)}$ and $\mathsf{Hyb}_{8,0}^{(b)}$ are identical. Furthermore, we have the following lemma for all $i \in [Q_{\mathrm{ro}}]$ which is proven below.

Lemma 5.9. *If Π' is q-bounded message hiding, then $\mathsf{Hyb}_{8,i-1}^{(b)} \approx_c \mathsf{Hyb}_{8,i}^{(b)}$.*

- $\mathsf{Hyb}_9^{(b)}$: This game is the same as $\mathsf{Hyb}_{8,Q_{\mathrm{ro}}}^{(b)}$ except that in responses to key queries $\mathcal{O}\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid})$, the challenger sets

$$\mathbf{v}'_{\mathsf{gid}} \coloneqq (0, \mathbf{s}) + \mathbf{r}_{\mathsf{gid}}(\mathbf{0}_k^\mathsf{T}, \mathbf{T} + \mathbf{T}') + \not{\sigma} \cdot \mathbf{y}_{\mathsf{gid}} \cdot \mathbf{B}_0^\mathsf{T}.$$

[13] More precisely, the challenger maintains a dictionary D which is initially empty. Upon receiving a random oracle query for $\mathsf{gid} \in \mathcal{GID}$, it samples a vector $\mathbf{y}_{\mathsf{gid}} \leftarrow\!\!\$\ \mathbb{Z}_p^d$ and sets $D[\mathsf{gid}] \coloneqq \mathbf{y}_{\mathsf{gid}}$. Upon receiving a key generation query $\mathcal{O}\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid})$, the challenger first retrieves the value $D[\mathsf{gid}]$ before responding to the query.

We have $\mathsf{Hyb}_{8,Q_{\mathrm{ro}}}^{(b)} \approx_s \mathsf{Hyb}_9^{(b)}$ by the same argument as in the transition $\mathsf{Hyb}_6^{(b)} \approx_s \mathsf{Hyb}_7^{(b)}$. Furthermore, we note that in $\mathsf{Hyb}_9^{(b)}$, all responses to oracle queries are independent of σ, so $[\![\sigma]\!]_{\mathrm{t}}$ perfectly hides $[\![\mu_b]\!]_{\mathrm{t}}$ and we conclude that $\mathsf{Hyb}_9^{(0)} \equiv \mathsf{Hyb}_9^{(1)}$.

□

Proof of Lemma 5.9. We consider a sequence of sub-hybrids $\overline{\mathsf{Hyb}}_0^{(b)}, \ldots, \overline{\mathsf{Hyb}}_3^{(b)}$.

- $\overline{\mathsf{Hyb}}_0^{(b)}$: This is $\mathsf{Hyb}_{8,i-1}^{(b)}$. In particular, upon receiving a key query $O\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid}_i)$, the challenger sets

$$\mathbf{v}'_{\mathsf{gid}_i} \coloneqq \underbrace{(0, \mathbf{s}) + \mathbf{r}_{\mathsf{gid}_i}(\mathbf{0}_k^{\mathrm{T}}, \mathbf{T} + \mathbf{T}')}_{=\boldsymbol{\alpha}_i} + \underbrace{\sigma \cdot \big((1, \mathbf{s}') + \mathbf{y}_{\mathsf{gid}_i} \cdot \mathbf{B}_0^{\mathrm{T}}\big)}_{=\boldsymbol{\beta}_i}$$

- $\overline{\mathsf{Hyb}}_1^{(b)}$: This is the same as $\overline{\mathsf{Hyb}}_0^{(b)}$ except for the following modifications.
  - Before receiving the first oracle query, the challenger initializes an empty set $\mathcal{H}_{\mathrm{pre}}$.
  - During the first query phase, upon $\mathcal{A}$ submitting a query $O\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid}_i)$, the challenger adds id to $\mathcal{H}_{\mathrm{pre}}$ and outputs

$$\mathsf{sk}_{\mathsf{id},\mathsf{gid}_i} \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}_{\mathsf{id}}, \mathsf{gid}, [\![\mathbf{v}]\!]_2 = [\![(1, \mathbf{r}_{\mathsf{gid}}, \boldsymbol{\alpha}_i, 0, 1)]\!]_2\big).$$

    Other oracle queries are handled as in $\overline{\mathsf{Hyb}}_0^{(b)}$.
  - For the challenge ciphertext $\mathsf{ct} = ([\![z]\!]_{\mathrm{t}}, \{\mathsf{ct}_j\}_{j\in[\ell]}, \mathbf{M}, \rho)$, the challenger computes

$$\mathsf{ct}_j \leftarrow \begin{cases} \mathsf{EncSK}'\big(\mathsf{msk}_{\rho(j)}, \mathbf{1}, [\![\mathbf{u}_j]\!]_1 = [\![(\mathbf{0}_{k+1}, \mathbf{M}_j^{\mathrm{T}}, 0, \boldsymbol{\beta}_i \cdot \mathbf{M}_j)]\!]_1\big) & \text{if } \rho(j) \in A \\ \mathsf{EncSK}'\big(\mathsf{msk}_{\mathsf{id}^*}, f, [\![\mathbf{u}_j]\!]_1 = [\![(\mathbf{0}_{k+1}, \mathbf{M}_j^{\mathrm{T}}, 0, \boldsymbol{\beta}_i \cdot \mathbf{M}_j)]\!]_1\big) & \text{if } \rho(j) = \mathsf{id}^*. \end{cases}$$

    for all $j \in [\ell]$ such that $\rho(j) \in \mathcal{H}_{\mathrm{pre}}$, and

$$\mathsf{ct}_j \leftarrow \begin{cases} \mathsf{EncSK}'\big(\mathsf{msk}_{\rho(j)}, \mathbf{1}, [\![\mathbf{u}_j]\!]_1 = [\![(\mathbf{0}_{k+1}, \mathbf{M}_j^{\mathrm{T}}, 1, 0)]\!]_1\big) & \text{if } \rho(j) \in A \\ \mathsf{EncSK}'\big(\mathsf{msk}_{\mathsf{id}^*}, f, [\![\mathbf{u}_j]\!]_1 = [\![(\mathbf{0}_{k+1}, \mathbf{M}_j^{\mathrm{T}}, 1, 0)]\!]_1\big) & \text{if } \rho(j) = \mathsf{id}^*. \end{cases}$$

    for all $j \in [\ell]$ such that $\rho(j) \in \mathcal{H}_{\mathrm{post}} \coloneqq (A \cap \mathcal{H}) \setminus \mathcal{H}_{\mathrm{pre}}$. The remaining ciphertext components are computed as in the previous hybrid.
  - During the second query phase, upon $\mathcal{A}$ submitting a query of the form $O\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid}_i)$ such that $\mathsf{id} \in \mathcal{H}_{\mathrm{post}}$, the challenger outputs

$$\mathsf{sk}_{\mathsf{id},\mathsf{gid}_i} \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}_{\mathsf{id}}, \mathsf{gid}, [\![\mathbf{v}]\!]_2 = [\![(1, \mathbf{r}_{\mathsf{gid}}, \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i \cdot \mathbf{M}_{\rho^{-1}(\mathsf{id})}, 0, 0)]\!]_2\big).$$

    Other oracle queries are handled as in $\overline{\mathsf{Hyb}}_0^{(b)}$.

  We note that the inner products between message and function vectors do not change from $\overline{\mathsf{Hyb}}_0^{(b)}$ to $\overline{\mathsf{Hyb}}_1^{(b)}$, in which case the 0-bounded message hiding of Π' implies that $\overline{\mathsf{Hyb}}_0^{(b)} \approx_c \overline{\mathsf{Hyb}}_1^{(b)}$.
- $\overline{\mathsf{Hyb}}_2^{(b)}$: This is the same as $\overline{\mathsf{Hyb}}_1^{(b)}$ except that the challenger replaces $\boldsymbol{\beta}_i$ with $\boldsymbol{\beta}'_i$, where

$$\begin{aligned} &\text{in } \overline{\mathsf{Hyb}}_1^{(b)}: && \boldsymbol{\beta}_i \coloneqq \sigma \cdot \big((1, \mathbf{s}') + \mathbf{y}_{\mathsf{gid}_i} \cdot \mathbf{B}_0^{\mathrm{T}}\big) \\ &\text{in } \overline{\mathsf{Hyb}}_2^{(b)}: && \boldsymbol{\beta}'_i \coloneqq \sigma \cdot \big(\cancel{(1, \mathbf{s}')} + \mathbf{y}_{\mathsf{gid}_i} \cdot \mathbf{B}_0^{\mathrm{T}}\big). \end{aligned}$$

  We have $\overline{\mathsf{Hyb}}_1^{(b)} \equiv \overline{\mathsf{Hyb}}_2^{(b)}$ which can be seen as follows. Pick any $\mathbf{w}_{\mathsf{gid}_i} = (1, \mathbf{s}'_{\mathsf{gid}_i}) \in A^{\perp}_{\mathsf{gid}_i}$ and recall that $\mathbf{w}_{\mathsf{gid}_i} \cdot \mathbf{M}_j = 0$ for all $j \in [\ell]$ such that $\rho(j) \in \mathcal{I}[\mathsf{gid}_i]$. In particular, $\mathbf{w}_{\mathsf{gid}_i}$ is orthogonal to all columns of $\mathbf{M}$ labeled by authorities in $C \cup C'$, so $\mathbf{w}_{\mathsf{gid}_i} \in A^{\perp}_{\mathrm{cor}}$. Combined with $\mathbf{w} \coloneqq (1, \mathbf{s}') \in A^{\perp}_{\mathrm{cor}}$, this gives $\mathbf{w}_{\mathsf{gid}_i} - \mathbf{w} \in M^{\perp}_{\mathrm{cor},0}$. Since the columns of $\mathbf{B}_0$ form a basis of $M^{\perp}_{\mathrm{cor},0}$, there exists $\mathbf{d}_i \in \mathbb{Z}_p^d$ such

that $\mathbf{w}_{\mathrm{gid}_i} = \mathbf{w} + \mathbf{d}_i\mathbf{B}_0^{\mathrm{T}}$. Note that the witnesses $\mathbf{w}_{\mathrm{gid}_i}$ and $\mathbf{d}_i$ are used solely in the analysis. Once the exposed set $\mathcal{I}[\mathrm{gid}_i]$ and the challenge data $(\mathbf{M}, \rho, C, C')$ are fixed, these witnesses are independent of $\mathbf{y}_{\mathrm{gid}_i}$; the simulator never computes or guesses them.

By the choice of $\mathbf{w}_{\mathrm{gid}_i}$, we have $\mathbf{w}_{\mathrm{gid}_i} \cdot \mathbf{M}_j = 0$ for every $j \in [\ell]$ such that $\rho(j) \in \mathcal{I}[\mathrm{gid}_i]$, and hence

$$(\mathbf{w} + \mathbf{y}_{\mathrm{gid}_i}\mathbf{B}_0^{\mathrm{T}}) \cdot \mathbf{M}_j = (\mathbf{w}_{\mathrm{gid}_i} + (\mathbf{y}_{\mathrm{gid}_i} - \mathbf{d}_i)\mathbf{B}_0^{\mathrm{T}}) \cdot \mathbf{M}_j = (\mathbf{y}_{\mathrm{gid}_i} - \mathbf{d}_i)\mathbf{B}_0^{\mathrm{T}} \cdot \mathbf{M}_j.$$

Viewing $\boldsymbol{\beta}_i = \boldsymbol{\beta}_i(\mathbf{y}_{\mathrm{gid}_i})$ and $\boldsymbol{\beta}'_i = \boldsymbol{\beta}'_i(\mathbf{y}_{\mathrm{gid}_i})$ as functions of the key randomness $\mathbf{y}_{\mathrm{gid}_i}$, we obtain

$$\begin{aligned}\boldsymbol{\beta}_i(\mathbf{y}_{\mathrm{gid}_i}) \cdot \mathbf{M}_j &= \sigma \cdot (\mathbf{w} + \mathbf{y}_{\mathrm{gid}_i}\mathbf{B}_0^{\mathrm{T}}) \cdot \mathbf{M}_j \\ &= \sigma \cdot (\mathbf{y}_{\mathrm{gid}_i} - \mathbf{d}_i)\mathbf{B}_0^{\mathrm{T}} \cdot \mathbf{M}_j \\ &= \boldsymbol{\beta}'_i(\mathbf{y}_{\mathrm{gid}_i} - \mathbf{d}_i) \cdot \mathbf{M}_j.\end{aligned}$$

The map $\mathbf{y}_{\mathrm{gid}_i} \to \mathbf{y}_{\mathrm{gid}_i} - \mathbf{d}_i$ is a bijection on $\mathbb{Z}_p^d$ and $\mathbf{y}_{\mathrm{gid}_i}$ is uniform. Therefore, by additionally noting that the vector $\mathbf{y}_{\mathrm{gid}_i}$ does not appear anywhere else in the simulation, we conclude that the following joint distributions are identical:

$$\left\{\boldsymbol{\beta}_i(\mathbf{y}_{\mathrm{gid}_i}) \cdot \mathbf{M}_j\right\}_{j \in \rho^{-1}(\mathcal{I}[\mathrm{gid}_i] \setminus (C \cup C'))} \equiv \left\{\boldsymbol{\beta}'_i(\mathbf{y}_{\mathrm{gid}_i}) \cdot \mathbf{M}_j\right\}_{j \in \rho^{-1}(\mathcal{I}[\mathrm{gid}_i] \setminus (C \cup C'))},$$

where the probability is taken over the choice of $\mathbf{y}_{\mathrm{gid}_i} \leftarrow\!\!\$\ \mathbb{Z}_p^d$.

- $\overline{\mathsf{Hyb}}_3^{(b)}$: This is $\mathsf{Hyb}_{8,i}^{(b)}$. We have $\overline{\mathsf{Hyb}}_2^{(b)} \approx_c \overline{\mathsf{Hyb}}_3^{(b)}$ which follows from a similar argument as the transition from $\overline{\mathsf{Hyb}}_0^{(b)}$ to $\overline{\mathsf{Hyb}}_1^{(b)}$.

□

5.3 Transformation to Decoder-Based Security

The three security notions from Definition 5.1 are formulated as standard indistinguishability games, in which the adversary $\mathcal{A}$ has to output a single guess of the challenge bit b. However, for traitor tracing we will need a strictly stronger flavour of these notions in which $\mathcal{A}$ instead outputs a (possibly stateful) pirate decoder D, and security holds whenever no such D has noticeable distinguishing advantage on freshly sampled challenge ciphertexts. This is because the natural reduction from a tracing adversary (Definition 6.1) to the underlying MA-2ABE security produces a decoder, and it is D—rather than the reduction itself—that has to distinguish between the two challenge distributions. We extend the predicate-hiding and message-hiding notions from Definition 5.1 to their decoder-based counterparts, in the spirit of [Zha25, Definition 20].

On the other hand, for mode indistinguishability, we adopt instead the *weak* decoder-based notion of [Zha25, Definition 43], which suffices for risky limited traitor tracing and is achievable from 0-bounded mode indistinguishability via a constant-overhead compiler (Theorem 5.15). The weak notion is parameterized by three constants $\alpha, \beta, \gamma \in (0, 1)$, whose roles are as follows. The thresholds α and β are lower bounds on the decoder's distinguishing advantage between encryptions of two adversarially chosen messages: α refers to ciphertexts produced by EncPK and β to ciphertexts produced by EncSK, where we explicitly allow $\beta < \alpha$. The constant γ is then a probability ratio relating the two regimes: rather than ruling out any decoder that distinguishes EncSK from EncPK ciphertexts, the weak notion only requires that the probability (over the coins of the experiment) that the decoder achieves advantage at least β on EncSK ciphertexts be at least γ times the probability that it achieves advantage at least α on EncPK ciphertexts.

The main differences between our decoder-based notions and those of [Zha25] are: (i) we operate in the multi-authority and attribute-based setting, so the adversary additionally has access to the oracles $O\mathsf{AuthSetup}$ and $O\mathsf{KeyGen}$, plus—for message hiding—the corruption oracle $O\mathsf{Corrupt}$; and (ii) the structural constraints—the special authority id^*, the access structure $\mathbb{A}$, and the corrupted set C—are inherited from the corresponding indistinguishability-based experiments of Figure 6.

Definition 5.10 ((Weak) decoder-based MA-2ABE security). Let Π be an MA-2ABE scheme as in Definition 5.1, and let $\epsilon = \epsilon(\lambda)$ be an inverse polynomial. We define the following notions of (weak) decoder-based security.

- **(Weak) decoder-based mode indistinguishability:** We say that Π is decoder-based mode indistinguishable if for all PPT adversaries $\mathcal{A}$ and all inverse polynomials ϵ, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\Pr\left[\mathsf{Good}_{\epsilon}^{\mathsf{mode}} \mid \mathsf{ExpDecMode}_{\Pi,\mathcal{A}}(1^{\lambda})\right] \leq \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{ExpDecMode}_{\Pi,\mathcal{A}}$ is defined in Figure 7 and $\mathsf{Good}_{\epsilon}^{\mathsf{mode}}$ is the event

$$\left|\Pr[D(\mathsf{ct}_0) = 1] - \Pr[D(\mathsf{ct}_1) = 1]\right| \geq \epsilon(\lambda),$$

with $\mathsf{ct}_0 \leftarrow \mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu)$ and $\mathsf{ct}_1 \leftarrow \mathsf{EncPK}(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu)$ encrypting the same message μ under the secret-key and public-key modes, respectively. The adversary has access to the oracles $O \coloneqq \{O\mathsf{AuthSetup}, O\mathsf{KeyGen}\}$.

We further say that Π is *weakly* decoder-based mode indistinguishable if there exist constants $\alpha, \beta, \gamma \in (0,1)$ such that for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\Pr\left[\mathsf{Good}_{\beta}^{\mathsf{sk}} \mid \mathsf{ExpWeakDecMode}_{\Pi,\mathcal{A}}(1^{\lambda})\right] \geq \gamma \cdot \Pr\left[\mathsf{Good}_{\alpha}^{\mathsf{pk}} \mid \mathsf{ExpWeakDecMode}_{\Pi,\mathcal{A}}(1^{\lambda})\right] - \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{ExpWeakDecMode}_{\Pi,\mathcal{A}}$ is defined in Figure 7, the adversary has access to the oracles $O \coloneqq \{O\mathsf{AuthSetup}, O\mathsf{KeyGen}\}$, and the events $\mathsf{Good}_{\alpha}^{\mathsf{pk}}, \mathsf{Good}_{\beta}^{\mathsf{sk}}$ are defined as follows. Let

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{pk}}(D) \coloneqq \Big|&\Pr\left[D(\mathsf{ct}) = 1 \mid \mathsf{ct} \leftarrow \mathsf{EncPK}(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_0)\right] \\
&- \Pr\left[D(\mathsf{ct}) = 1 \mid \mathsf{ct} \leftarrow \mathsf{EncPK}(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_1)\right]\Big|, \\
\mathsf{Adv}^{\mathsf{sk}}(D) \coloneqq \Big|&\Pr\left[D(\mathsf{ct}) = 1 \mid \mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_0)\right] \\
&- \Pr\left[D(\mathsf{ct}) = 1 \mid \mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_1)\right]\Big|,
\end{aligned}$$

denote D's distinguishing advantage between encryptions of μ_0 and μ_1 under the public-key and secret-key modes, respectively. Then, $\mathsf{Good}_{\alpha}^{\mathsf{pk}}$ is the event $\mathsf{Adv}^{\mathsf{pk}}(D) \geq \alpha$ and $\mathsf{Good}_{\beta}^{\mathsf{sk}}$ is the event $\mathsf{Adv}^{\mathsf{sk}}(D) \geq \beta$.

- **Decoder-based predicate hiding:** We say that Π is decoder-based predicate hiding if for all PPT adversaries $\mathcal{A}$ and all inverse polynomials ϵ, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\Pr\left[\mathsf{Good}_{\epsilon}^{\mathsf{pred}} \mid \mathsf{ExpDecPred}_{\Pi,\mathcal{A}}(1^{\lambda})\right] \leq \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{ExpDecPred}_{\Pi,\mathcal{A}}$ is defined in Figure 7 and $\mathsf{Good}_{\epsilon}^{\mathsf{pred}}$ is the event

$$\left|\Pr[D(\mathsf{ct}_0) = 1] - \Pr[D(\mathsf{ct}_1) = 1]\right| \geq \epsilon(\lambda),$$

with $\mathsf{ct}_b \leftarrow \mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi_b = (f_b, \mathbb{A}), \mu)$ for $b \in \{0,1\}$. The adversary has access to the oracles $O \coloneqq \{O\mathsf{AuthSetup}, O\mathsf{KeyGen}\}$.

- **Decoder-based message hiding:** We say that Π is decoder-based message hiding if for all PPT adversaries $\mathcal{A}$ and all inverse polynomials ϵ, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\Pr\left[\mathsf{Good}_{\epsilon}^{\mathsf{msg}} \mid \mathsf{ExpDecMsg}_{\Pi,\mathcal{A}}(1^{\lambda})\right] \leq \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{ExpDecMsg}_{\Pi,\mathcal{A}}$ is defined in Figure 7 and $\mathsf{Good}_{\epsilon}^{\mathsf{msg}}$ is the event

$$\left|\Pr[D(\mathsf{ct}_0) = 1] - \Pr[D(\mathsf{ct}_1) = 1]\right| \geq \epsilon(\lambda),$$

with $\mathsf{ct}_b \leftarrow \mathsf{EncSK}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_b)$ for $b \in \{0,1\}$. The adversary has access to the oracles $O \coloneqq \{O\mathsf{AuthSetup}, O\mathsf{KeyGen}, O\mathsf{Corrupt}\}$, and we set $C' \coloneqq A \setminus \mathcal{K}$.

$\mathsf{Exp}\boxed{\mathsf{Weak}}\mathsf{DecMode}_{\Pi,\mathcal{A}}(1^\lambda)$

$f^* \leftarrow \mathcal{A}(1^\lambda)$
$\mathcal{K}, Q \coloneqq [\cdot]$
$\mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$
$(D, \mathsf{id}^*, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \mathbb{A}, \boxed{\mu}\ \boxed{\mu_0, \mu_1}) \leftarrow \mathcal{A}^{O}(\mathsf{gp})$
assert $\mathsf{id}^* \notin A \wedge \mathcal{K}[\mathsf{id}^*] \neq \bot$
// $\mathcal{K}[\mathsf{id}^*] = (\mathsf{pk}_{\mathsf{id}^*}, \mathsf{msk}_{\mathsf{id}^*})$
return $(D, \mathcal{K}[\mathsf{id}^*], \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi = (f^*, \mathbb{A}), \boxed{\mu}\ \boxed{\mu_0, \mu_1})$

$\mathsf{ExpDecPred}_{\Pi,\mathcal{A}}(1^\lambda)$

$(f_0, f_1) \leftarrow \mathcal{A}(1^\lambda)$
$\mathcal{K}, Q \coloneqq [\cdot]$
$\mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$
$(D, \mathsf{id}^*, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \mathbb{A}, \mu) \leftarrow \mathcal{A}^{O}(\mathsf{gp})$
assert $\mathsf{id}^* \notin A \wedge \mathcal{K}[\mathsf{id}^*] \neq \bot$
parse $(\mathsf{pk}_{\mathsf{id}^*}, \mathsf{msk}_{\mathsf{id}^*}) \coloneqq \mathcal{K}[\mathsf{id}^*]$
assert $\forall \mathsf{gid} \in Q[\mathsf{id}^*],\ f_0(\mathsf{gid}) = f_1(\mathsf{gid})$
return $(D, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, f_0, f_1, \mathbb{A}, \mu)$

$O\mathsf{AuthSetup}(\mathsf{id})$

assert $\mathcal{K}[\mathsf{id}] = \bot$
$(\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}}) \leftarrow \mathsf{AuthSetup}(\mathsf{gp}, \mathsf{id})$
$\mathcal{K}[\mathsf{id}] \coloneqq (\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}})$
return $\mathsf{pk}_{\mathsf{id}}$

$\mathsf{ExpDecMsg}_{\Pi,\mathcal{A}}(1^\lambda)$

$f^* \leftarrow \mathcal{A}(1^\lambda)$
$\mathcal{I}, \mathcal{K}, Q \coloneqq [\cdot];\ C \coloneqq \emptyset$
$\mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$
$(D, \mathsf{id}^*, A, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in C'}, \mathbb{A}, \mu_0, \mu_1) \leftarrow \mathcal{A}^{O}(\mathsf{gp})$
assert $\mathsf{id}^* \notin (A \cup C) \wedge \mathcal{K}[\mathsf{id}^*] \neq \bot$
$\forall \mathsf{id} \in (A \setminus C') \cup \{\mathsf{id}^*\},$ **parse** $(\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}}) \coloneqq \mathcal{K}[\mathsf{id}]$
$\forall \mathsf{gid},\ \mathcal{I}[\mathsf{gid}] \coloneqq \{\mathsf{id} : \mathsf{gid} \in Q[\mathsf{id}]\} \cup C \cup C'$
assert $(C \cup C') \notin \mathbb{A} \wedge \big[\forall \mathsf{gid} \in \bigcup_{\mathsf{id}\in A\cup\{\mathsf{id}^*\}} Q[\mathsf{id}],$
  $\mathcal{I}[\mathsf{gid}] \notin \mathbb{A} \vee \big(f^*(\mathsf{gid}) = 0 \wedge \mathcal{I}[\mathsf{gid}] \setminus \{\mathsf{id}^*\} \notin \mathbb{A}\big)\big]$
return $(D, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi = (f^*, \mathbb{A}), \mu_0, \mu_1)$

$O\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid})$

assert $\mathcal{K}[\mathsf{id}] \neq \bot$
if $Q[\mathsf{id}] = \bot$ **then** $Q[\mathsf{id}] \coloneqq \emptyset$
$Q[\mathsf{id}] \coloneqq Q[\mathsf{id}] \cup \{\mathsf{gid}\}$
parse $(\mathsf{pk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}}) \coloneqq \mathcal{K}[\mathsf{id}]$
return $\mathsf{sk}_{\mathsf{id},\mathsf{gid}} \leftarrow \mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}}, \mathsf{gid})$

$O\mathsf{Corrupt}(\mathsf{id})$

assert $\mathcal{K}[\mathsf{id}] \neq \bot$
$C \coloneqq C \cup \{\mathsf{id}\}$
return $\mathcal{K}[\mathsf{id}]$

Figure 7: Experiments $\mathsf{ExpWeakDecMode}_{\Pi,\mathcal{A}}$, $\mathsf{ExpDecMode}_{\Pi,\mathcal{A}}$, $\mathsf{ExpDecPred}_{\Pi,\mathcal{A}}$ and $\mathsf{ExpDecMsg}_{\Pi,\mathcal{A}}$ for Definition 5.10. Each experiment outputs the adversary's choice of challenge parameters together with the decoder D, which are then fed into the corresponding events. $\mathsf{ExpDecMode}_{\Pi,\mathcal{A}}$ uses a single message denoted with the solid box $\boxed{\mu}$, whereas its weak variant $\mathsf{ExpWeakDecMode}_{\Pi,\mathcal{A}}$ replaces it with a pair of messages denoted with the gray box μ_0, μ_1. All other lines are common to both games.

The inner probabilities defining each event are taken solely over the coins of EncSK/EncPK and D, while the outer probability defining the security advantage is taken over the coins of GlobalSetup, the oracles, and the adversary $\mathcal{A}$.

We observe that the decoder-based notions from Definition 5.10 are already implied by their 1-bounded indistinguishability-based counterparts from Definition 5.1, with no additional machinery. This adapts the argument of Goyal, Koppula, and Waters [GKW18, Section 4.1], who showed that 1-bounded security implies the corresponding decoder-based security for mixed FE, to our MA-2ABE setting. That is, a single OEncSK query suffices—even though the decoder must be tested against *two* secret-key challenge ciphertexts—owing to the "compare-two-evaluations" trick of [GKW18]: the reduction guesses which of the two challenge functions (resp. messages) to encrypt under its single oracle query, and then tests whether the decoder agrees on that oracle ciphertext and challenge ciphertext. Therefore, we derive the following theorems, whose proofs are slight adaptations of the ones given in [GKW18, Section 4.1], and hence, we omit the proofs here.

Theorem 5.11 (Adapted from [GKW18, Lemma 4.1 and Lemma 6.1]). *If an MA-2ABE scheme Π is 1-bounded mode indistinguishable (Definition 5.1), then it is decoder-based mode indistinguishable (Definition 5.10).*[14]

Theorem 5.12 (Adapted from [GKW18, Lemma 4.2 and Lemma 6.2]). *If an MA-2ABE scheme Π is 1-bounded predicate hiding (Definition 5.1), then it is decoder-based predicate hiding (Definition 5.10).*[15]

[14] Our *mode indistinguishability* property for MA-2ABE corresponds to the *accept indistinguishability* property for mixed FE given in [GKW18].

[15] Our *predicate-hiding* property for MA-2ABE corresponds to the *function-hiding* property for mixed FE given in [GKW18].

Theorem 5.13 (Adapted from [GKW18, Lemma 4.3 and Lemma 6.3]). *If an MA-2ABE scheme Π is 1-bounded message hiding (Definition 5.1), then it is decoder-based message hiding (Definition 5.10).*

Weak decoder-based mode indistinguishability. Next, we turn to *weak* decoder-based mode indistinguishability, and adapt [Zha25, Theorem 44] to our multi-authority attribute-based setting. That is, we prove that any 0-bounded mode-indistinguishable MA-2ABE scheme can be generically transformed into one satisfying weak decoder-based mode indistinguishability, at a constant overhead in parameters and running time, while preserving q-bounded predicate-hiding and message-hiding. The compiler runs ℓ independent copies of the underlying scheme in parallel, where ℓ is a sufficiently large constant.

Construction 5.14 (Compiler for weak decoder-based mode indistinguishability). Fix constants $\alpha, \beta, \gamma \in (0,1)$ and an integer $\ell \in \mathbb{N}$ satisfying $\beta < \alpha$ and

$$\ell(1-\gamma)(\alpha-\beta)^2 > 4.$$

For example, we can take $\ell = 5$, $\alpha = 19/20$, $\beta = 1/20$, $\gamma = 1/162$ as in [Zha25].

Given any MA-2ABE scheme $\Pi = (\mathsf{GlobalSetup}, \mathsf{AuthSetup}, \mathsf{KeyGen}, \mathsf{EncPK}, \mathsf{EncSK}, \mathsf{Dec})$, define the compiled scheme $\Pi' = (\mathsf{GlobalSetup}', \mathsf{AuthSetup}', \mathsf{KeyGen}', \mathsf{EncPK}', \mathsf{EncSK}', \mathsf{Dec}')$ by running ℓ fully independent copies of Π in parallel—each copy with its own freshly sampled global parameters, mirroring the riskiness mitigation compiler of Construction 6.8.

- $\mathsf{GlobalSetup}'(1^\lambda)$ for each $\zeta \in [\ell]$, runs $\mathsf{gp}_\zeta \leftarrow \mathsf{GlobalSetup}(1^\lambda)$ and returns $\mathsf{gp} := (\mathsf{gp}_\zeta)_{\zeta\in[\ell]}$.
- $\mathsf{AuthSetup}'(\mathsf{gp}, \mathsf{id})$ parses $\mathsf{gp} = (\mathsf{gp}_\zeta)_{\zeta\in[\ell]}$, for each $\zeta \in [\ell]$, runs $(\mathsf{pk}_{\mathsf{id}}^\zeta, \mathsf{msk}_{\mathsf{id}}^\zeta) \leftarrow \mathsf{AuthSetup}(\mathsf{gp}_\zeta, \mathsf{id})$ and returns $\mathsf{pk}'_{\mathsf{id}} := (\mathsf{pk}_{\mathsf{id}}^\zeta)_{\zeta\in[\ell]}$ and $\mathsf{msk}'_{\mathsf{id}} := (\mathsf{msk}_{\mathsf{id}}^\zeta)_{\zeta\in[\ell]}$.
- $\mathsf{KeyGen}'(\mathsf{gp}, \mathsf{msk}'_{\mathsf{id}}, \mathsf{gid})$ parses $\mathsf{gp} = (\mathsf{gp}_\zeta)_{\zeta\in[\ell]}$ and $\mathsf{msk}'_{\mathsf{id}} = (\mathsf{msk}_{\mathsf{id}}^\zeta)_{\zeta\in[\ell]}$, for each $\zeta \in [\ell]$, runs $\mathsf{sk}_{\mathsf{id},\mathsf{gid}}^\zeta \leftarrow \mathsf{KeyGen}(\mathsf{gp}_\zeta, \mathsf{msk}_{\mathsf{id}}^\zeta, \mathsf{gid})$ and returns $\mathsf{sk}'_{\mathsf{id},\mathsf{gid}} := (\mathsf{sk}_{\mathsf{id},\mathsf{gid}}^\zeta)_{\zeta\in[\ell]}$.
- $\mathsf{EncPK}'(\mathsf{gp}, \mathsf{pk}'_{\mathsf{id}^*}, \{\mathsf{pk}'_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu)$ parses $\mathsf{gp} = (\mathsf{gp}_\zeta)_{\zeta\in[\ell]}$ and $\mathsf{pk}'_{\mathsf{id}} = (\mathsf{pk}_{\mathsf{id}}^{\zeta'})_{\zeta'\in[\ell]}$, for each $\mathsf{id} \in A \cup \{\mathsf{id}^*\}$, samples $\zeta \leftarrow [\ell]$, runs $\mathsf{ct} \leftarrow \mathsf{EncPK}(\mathsf{gp}_\zeta, \mathsf{pk}_{\mathsf{id}^*}^\zeta, \{\mathsf{pk}_{\mathsf{id}}^\zeta\}_{\mathsf{id}\in A}, \pi, \mu)$, and returns $\mathsf{ct}' := (\zeta, \mathsf{ct})$.
- $\mathsf{EncSK}'(\mathsf{gp}, \mathsf{msk}'_{\mathsf{id}^*}, \{\mathsf{pk}'_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu)$ parses $\mathsf{gp}, \mathsf{msk}'_{\mathsf{id}^*}$ and $\{\mathsf{pk}'_{\mathsf{id}}\}_{\mathsf{id}\in A}$ as above, samples $\zeta \leftarrow [\ell]$, runs $\mathsf{ct} \leftarrow \mathsf{EncSK}(\mathsf{gp}_\zeta, \mathsf{msk}_{\mathsf{id}^*}^\zeta, \{\mathsf{pk}_{\mathsf{id}}^\zeta\}_{\mathsf{id}\in A}, \pi, \mu)$, and returns $\mathsf{ct}' := (\zeta, \mathsf{ct})$.
- $\mathsf{Dec}'(\mathsf{gp}, \{\mathsf{sk}'_{\mathsf{id},\mathsf{gid}}\}_{\mathsf{id}\in S}, \mathsf{ct}')$: parses $\mathsf{gp} = (\mathsf{gp}_\zeta)_{\zeta\in[\ell]}$, $\mathsf{ct}' = (\zeta, \mathsf{ct})$, and $\mathsf{sk}'_{\mathsf{id},\mathsf{gid}} = (\mathsf{sk}_{\mathsf{id},\mathsf{gid}}^{\zeta'})_{\zeta'\in[\ell]}$ for each $\mathsf{id} \in S$, and returns $\mathsf{Dec}(\mathsf{gp}_\zeta, \{\mathsf{sk}_{\mathsf{id},\mathsf{gid}}^\zeta\}_{\mathsf{id}\in S}, \mathsf{ct})$.

Theorem 5.15 (Adapted from [Zha25, Theorem 44]). *Let Π be a 0-bounded mode-indistinguishable MA-2ABE scheme as in Definition 5.1. Then, the compiled scheme Π' of Construction 5.14 is weak decoder-based mode indistinguishable (in the sense of Definition 5.10) with the constants α, β, γ chosen in the construction.*

Proof. Correctness of Π' is immediate: the index $\zeta \in [\ell]$ packaged in ct' singles out a coherent underlying instance of Π (with its own gp_ζ), so Dec' reduces to a valid invocation of Dec. The preservation of q-bounded predicate-hiding and q-bounded message-hiding is obtained by a standard "guess-the-index" simulation: a reduction $\mathcal{B}$ against the corresponding q-bounded indistinguishability-based notion of Π samples a target instance $\zeta^* \leftarrow [\ell]$ a priori, embeds Π's global parameters gp as gp_{ζ^*}, samples the remaining $\mathsf{gp}_{\zeta'} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$ for $\zeta' \neq \zeta^*$ locally, and runs all $\ell - 1$ non-challenge instances of $\mathsf{AuthSetup}, \mathsf{KeyGen}$ locally (so that it can answer all $O\mathsf{AuthSetup}$, $O\mathsf{KeyGen}$, and $O\mathsf{Corrupt}$ queries by itself). Then, it forwards Π's challenge ciphertext as the ζ^*-th component of Π''s challenge, and answers each $O\mathsf{EncSK}$ query of the adversary either by forwarding it to Π's oracle (if the freshly sampled ζ for that query coincides with ζ^*) or by encrypting locally otherwise. Since each adversarial $O\mathsf{EncSK}$ query results in at most one query to Π's oracle, the q-bound is preserved.

We now turn to the main claim: weak decoder-based mode indistinguishability of Π'. Fix a PPT adversary $\mathcal{A}$ against this notion and set

$$\tau(\lambda) := \Pr\left[\mathsf{Good}_\alpha^{\mathsf{pk}} \mid \mathsf{ExpWeakDecMode}_{\Pi',\mathcal{A}}(1^\lambda)\right],$$
$$\sigma(\lambda) := \Pr\left[\mathsf{Good}_\beta^{\mathsf{sk}} \mid \mathsf{ExpWeakDecMode}_{\Pi',\mathcal{A}}(1^\lambda)\right].$$

We must show that $\delta(\lambda) \coloneqq \gamma \cdot \tau(\lambda) - \sigma(\lambda)$ is negligible. For each $\zeta \in [\ell]$, define

$$p_\zeta \coloneqq \Pr\left[D((\zeta, \mathrm{ct})) = m \;\middle|\; m \leftarrow \{0,1\},\ \mathrm{ct} \leftarrow \mathsf{EncSK}(\mathsf{gp}_\zeta, \mathsf{msk}^{\zeta}_{\mathsf{id}^*}, \{\mathsf{pk}^{\zeta}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_m)\right],$$
$$q_\zeta \coloneqq \Pr\left[D((\zeta, \mathrm{ct})) = m \;\middle|\; m \leftarrow \{0,1\},\ \mathrm{ct} \leftarrow \mathsf{EncPK}(\mathsf{gp}_\zeta, \mathsf{pk}^{\zeta}_{\mathsf{id}^*}, \{\mathsf{pk}^{\zeta}_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_m)\right],$$

so that $p_\zeta, q_\zeta \in [0,1]$ measure the decoder's per-instance distinguishing advantage between encryptions of μ_0 and μ_1 under EncSK and EncPK, respectively. A short computation shows that $\mathsf{Adv}^{\mathrm{sk}}(D) = |2p-1|$ and $\mathsf{Adv}^{\mathrm{pk}}(D) = |2q-1|$, where $p \coloneqq \frac{1}{\ell}\sum_\zeta p_\zeta$ and $q \coloneqq \frac{1}{\ell}\sum_\zeta q_\zeta$. By symmetry (swapping μ_0 and μ_1 if necessary), we may assume throughout that the signed quantities $2p-1$ and $2q-1$ are non-negative, so that $\mathsf{Good}^{\mathrm{pk}}_\alpha$ is the event $2q-1 \geq \alpha$ and $\mathsf{Good}^{\mathrm{sk}}_\beta$ is the event $2p-1 \geq \beta$. Set $\Delta_\zeta \coloneqq p_\zeta - q_\zeta \in [-1,1]$ and $\Delta \coloneqq \frac{1}{\ell}\sum_\zeta \Delta_\zeta = p - q$.

The technical heart of the argument is the following bound on the cross-correlation between distinct one-instance advantages, which corresponds to the multi-authority analog of [Zha25, Lemma 47].

Lemma 5.16. *Let* Ev *be any event efficiently recognizable from the inputs and outputs of* $\mathcal{A}$*. If* Π *is* 0*-bounded mode indistinguishable, then for all distinct* $i, j \in [\ell]$ *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,*

$$|\mathbb{E}[\Delta_i \Delta_j \mid \mathsf{Ev}]| \cdot \Pr[\mathsf{Ev}] \leq \mathsf{negl}(\lambda).$$

Proof of Lemma 5.16. The overall structure of the reduction is: use Π's challenger for instance i, simulate instance j locally, and XOR the outputs of D, which is similar to the proof of [Zha25, Lemma 47]. Concretely, the reduction $\mathcal{B}^{(i,j)}$ proceeds as follows.

Setup and oracle simulation. On input gp_i from Π's challenger, $\mathcal{B}^{(i,j)}$ samples $\mathsf{gp}_{\zeta'} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$ for each $\zeta' \in [\ell] \setminus \{i\}$, sets $\mathsf{gp} \coloneqq (\mathsf{gp}_\zeta)_{\zeta\in[\ell]}$, and invokes $\mathcal{A}$ on gp. Each $\mathcal{O}\mathsf{AuthSetup}(\mathsf{id})$ query from $\mathcal{A}$ is answered by querying Π's $\mathcal{O}\mathsf{AuthSetup}(\mathsf{id})$ once (yielding $\mathsf{pk}^i_{\mathsf{id}}$ under gp_i) and running $\mathsf{AuthSetup}(\mathsf{gp}_{\zeta'}, \mathsf{id})$ locally for each $\zeta' \neq i$ (yielding $(\mathsf{pk}^{\zeta'}_{\mathsf{id}}, \mathsf{msk}^{\zeta'}_{\mathsf{id}})$). Finally, $\mathcal{B}$ returns $\mathsf{pk}'_{\mathsf{id}} = (\mathsf{pk}^{\zeta'}_{\mathsf{id}})_{\zeta'\in[\ell]}$. Each $\mathcal{O}\mathsf{KeyGen}(\mathsf{id}, \mathsf{gid})$ query is handled analogously.

Challenge phase. When $\mathcal{A}$ outputs $(\mathsf{id}^*, \{\mathsf{pk}'_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_0, \mu_1, D)$, $\mathcal{B}^{(i,j)}$ samples $m \leftarrow \{0,1\}$ and submits $(\mathsf{id}^*, \{\mathsf{pk}^i_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_m)$ to Π's mode-indistinguishability challenger. It receives a ciphertext c^* which is either $\mathsf{EncSK}(\mathsf{gp}_i, \mathsf{msk}^i_{\mathsf{id}^*}, \{\mathsf{pk}^i_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_m)$ (when Π's challenge bit is $b = 0$) or $\mathsf{EncPK}(\mathsf{gp}_i, \mathsf{pk}^i_{\mathsf{id}^*}, \{\mathsf{pk}^i_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_m)$ (when $b = 1$).

Decoder evaluation. $\mathcal{B}^{(i,j)}$ checks whether Ev holds. If not, it outputs a fresh uniform bit. Otherwise, it samples $m', \beta \leftarrow \{0,1\}$, computes

$$d_0 \leftarrow \mathsf{EncSK}(\mathsf{gp}_j, \mathsf{msk}^j_{\mathsf{id}^*}, \{\mathsf{pk}^j_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_{m'}),$$
$$d_1 \leftarrow \mathsf{EncPK}(\mathsf{gp}_j, \mathsf{pk}^j_{\mathsf{id}^*}, \{\mathsf{pk}^j_{\mathsf{id}}\}_{\mathsf{id}\in A}, \pi, \mu_{m'})$$

locally under gp_j, and outputs

$$b' \coloneqq \beta \oplus \big(m \oplus D((i, c^*))\big) \oplus \big(m' \oplus D((j, d_\beta))\big).$$

Since $\mathcal{B}^{(i,j)}$ never queries Π's $\mathcal{O}\mathsf{EncSK}$, it qualifies as a 0-bounded adversary.

Success probability. Conditioned on Ev, the bits $m \oplus D((i, c^*))$ and $m' \oplus D((j, d_\beta))$ are mutually independent and independent of β, with

$$\Pr[m \oplus D((i,c^*)) = 0 \mid b = 0, \mathsf{Ev}] = p_i, \qquad \Pr[m \oplus D((i,c^*)) = 0 \mid b = 1, \mathsf{Ev}] = q_i,$$
$$\Pr[m' \oplus D((j,d_\beta)) = 0 \mid \beta = 0, \mathsf{Ev}] = p_j, \qquad \Pr[m' \oplus D((j,d_\beta)) = 0 \mid \beta = 1, \mathsf{Ev}] = q_j.$$

These are precisely the probabilities driving the calculation of [Zha25, Lemma 47] and the same algebraic identity therefore yields

$$\Pr[b' = b] = \frac{1}{2} + \frac{1}{2}\mathbb{E}[\Delta_i \Delta_j \mid \mathsf{Ev}] \cdot \Pr[\mathsf{Ev}].$$

Since Π is 0-bounded mode indistinguishable, $|2\Pr[b' = b] - 1| \le \mathsf{negl}(\lambda)$, so $|\mathbb{E}[\Delta_i\Delta_j \mid \mathsf{Ev}]| \cdot \Pr[\mathsf{Ev}] \le \mathsf{negl}(\lambda)$, as claimed by the lemma. □

Conclusion of the theorem proof. We now mirror the proof of [Zha25, Theorem 44], with a sole adaptation of the empirical estimate of q being message-distinguishing rather than KEM-key-distinguishing.

Suppose for contradiction that $\delta(\lambda)$ is non-negligible. Then, there exist an inverse polynomial $T(\lambda)$ and an infinite set $\Lambda \subseteq \mathbb{N}$ such that $\delta(\lambda) \ge T(\lambda)$ for all $\lambda \in \Lambda$. In particular, $\tau(\lambda) \ge \delta(\lambda)/\gamma \ge T(\lambda)$. Pick perturbed constants α', γ' with $\beta < \alpha' < \alpha$ and $\gamma < \gamma' < 1$ such that $\ell(1-\gamma')(\alpha'-\beta)^2 > 4$. This is possible because $\ell(1-\gamma)(\alpha-\beta)^2 > 4$ holds strictly.

Define the event Ev as follows: the reduction draws $O(\lambda)$ fresh $\mathsf{EncPK}'(\cdot, \mu_0)$ and $\mathsf{EncPK}'(\cdot, \mu_1)$ ciphertexts, feeds them to D, and reads off the empirical message-distinguishing frequency $\tilde{q}$ of D on them. Then, Ev is the event that $2\tilde{q} - 1 \ge (\alpha + \alpha')/2$. By a Hoeffding bound, $|\tilde{q} - q| \le (\alpha - \alpha')/4$ except with probability $2^{-\lambda}$, so $\Pr[\mathsf{Ev}] \ge \tau(\lambda) - 2^{-\lambda}$ and $\Pr[2q - 1 < \alpha' \wedge \mathsf{Ev}] \le 2^{-\lambda}$.

With the event Ev in hand, recall that $\Delta = \frac{1}{\ell}\sum_\zeta \Delta_\zeta$ with each $\Delta_\zeta \in [-1, 1]$. Using the trivial bound $\Delta_\zeta^2 \le 1$ for the diagonal terms and Lemma 5.16 (applied to each pair $i \ne j$) for the cross terms, we obtain

$$\begin{aligned}
\mathbb{E}[\Delta^2 \mid \mathsf{Ev}] &= \frac{1}{\ell^2}\left(\sum_{\zeta\in[\ell]} \mathbb{E}[\Delta_\zeta^2 \mid \mathsf{Ev}] + \sum_{i\ne j} \mathbb{E}[\Delta_i\Delta_j \mid \mathsf{Ev}]\right) \\
&\le \frac{1}{\ell^2}\left(\ell + \sum_{i\ne j} \mathbb{E}[\Delta_i\Delta_j \mid \mathsf{Ev}]\right) \\
&\le \frac{1}{\ell} + \frac{\mathsf{negl}(\lambda)}{\Pr[\mathsf{Ev}]}.
\end{aligned}$$

Following [Zha25, Theorem 44], by Markov's inequality on $\mathbb{E}[\Delta^2 \mid \mathsf{Ev}]$ together with the lower bound $\Pr[\mathsf{Ev}] \ge T(\lambda)/2$ (valid for large enough $\lambda \in \Lambda$) yields $\Pr[\Delta < (\alpha' - \beta)/2 \wedge \mathsf{Ev}] \ge \gamma' \Pr[\mathsf{Ev}]$ by the choice of α', γ'. Combining this with $\Pr[2q - 1 < \alpha' \wedge \mathsf{Ev}] \le 2^{-\lambda}$ and $p = q + \Delta$ then gives $\sigma(\lambda) \ge \gamma\tau(\lambda)$ for $\lambda \in \Lambda$ large enough. This contradicts $\delta(\lambda) = \gamma\tau(\lambda) - \sigma(\lambda) \ge T(\lambda) > 0$, and hence, $\delta(\lambda)$ must in fact be negligible. This concludes the proof of the theorem. □

5.4 Special Case: Multi-Authority Embedded-Identity Private Linear Broadcast Encryption

In the construction of MA-AB-TT with full embedded identities (Definition 6.1), we will instantiate the identity predicate class $\mathcal{F} : \mathcal{GID} \to \{0, 1\}$ of Definition 5.1 with a particular two-parameter family of predicates that arises from *embedded-identity private linear broadcast encryption* (EIPLBE), introduced by [GKW19]. We call the resulting mixed MA-2ABE specialization MA-EIPLBE.

The EIPLBE function class. For positive integers $\kappa, \ell \in \mathbb{N}$, let $\mathcal{GID}_{\kappa,\ell} := [2^\kappa] \times \{0,1\}^\ell$, so that a global user identifier $\mathsf{gid} = (j, \mathsf{id})$ consists of an *index* $j \in [2^\kappa]$ (drawn from an exponentially large space) and an *embedded identity* $\mathsf{id} \in \{0,1\}^\ell$. The EIPLBE predicates are parameterized by a pair $(t, i) \in [0; 2^\kappa] \times [0; \ell]$, where t is a *threshold* on the index and i selects between an index-only predicate ($i = 0$) and a single bit of the embedded identity ($i > 0$). Concretely, the EIPLBE function class is

$$\mathcal{F}^{\mathrm{EIPLBE}}_{\kappa,\ell} := \left\{f_{t,i} : \mathcal{GID}_{\kappa,\ell} \to \{0,1\}\right\}_{(t,i)\in[0;2^\kappa]\times[0;\ell]},$$

where each $f_{t,i}$ is defined as

$$f_{t,i}\big((j, \mathsf{id})\big) := \begin{cases} \mathbf{1}\big[j \le t\big] & \text{if } i = 0, \\ \mathbf{1}\big[j < t \ \vee \ (j, \mathsf{id}_i) = (t, 1)\big] & \text{if } i > 0, \end{cases}$$

with id_i denoting the i-th bit of id. The two cases behave as follows.

- **Case $i = 0$:** The predicate $f_{t,0}$ depends only on the index part of gid and equals 1 whenever $j \le t$. As t ranges over $[0; 2^\kappa]$, the set of users satisfying $f_{t,0}(\mathsf{gid}) = 1$ grows monotonically from $\emptyset$ to all of $\mathcal{GID}_{\kappa,\ell}$.

This is what enables binary-search tracing [NWZ16] to localize the index j^* of an accused user.

- **Case $i > 0$:** The predicate $f_{t,i}$ satisfies $f_{t,i}(\mathsf{gid}) = f_{t,0}(\mathsf{gid})$ for every $\mathsf{gid} = (j, \mathsf{id})$ with $j \neq t$, but differs precisely on those gid with $j = t$, i.e., it evaluates to id_i rather than to 1. Once the index j^* of a traced user has been localized at $t = j^*$, querying $f_{j^*,i}$ for $i = 1, \ldots, \ell$ then reveals the corresponding embedded identity bit by bit.

MA-EIPLBE. We now specialize Definition 5.1 to this function class.

Definition 5.17 (MA-EIPLBE). Let $\kappa = \kappa(\lambda)$ and $\ell = \ell(\lambda)$ be polynomially bounded functions of the security parameter. A multi-authority attribute-based embedded-identity private linear broadcast encryption (MA-EIPLBE) scheme with parameters (κ, ℓ) is a mixed MA-2ABE scheme (Definition 5.1) for the global-identifier space $\mathcal{GID} \coloneqq \mathcal{GID}_{\kappa,\ell}$ and the function class $\mathcal{F} \coloneqq \mathcal{F}_{\kappa,\ell}^{\mathrm{EIPLBE}}$.

The correctness, q-bounded mode-indistinguishability, q-bounded predicate-hiding, and q-bounded message-hiding properties of MA-EIPLBE are inherited verbatim from Definition 5.1, instantiated with the above $\mathcal{GID}$ and $\mathcal{F}$. Analogously, the (weak) decoder-based variants of these notions (Definition 5.10) carry over verbatim.

Remark 5.18. Our MA-EIPLBE is the multi-authority analog of the EIPLBE primitive of [GKW19, Zha25]. More precisely, our definition (i) inherits the access-structure flexibility of mixed MA-2ABE, i.e., the ciphertexts can additionally enforce a monotone access structure $\mathbb{A} \subseteq 2^{\mathcal{ID}}$ over an arbitrary set of attribute authorities, and decryption requires both the EIPLBE predicate $f_{t,i}(\mathsf{gid}) = 1$ *and* a satisfying set of attribute keys for $\mathbb{A}$, and (ii) is parameterized by a tracing-authority identifier $\mathsf{id}^* \in \mathcal{ID}$ (whose master key is used to produce the secret-mode ciphertexts that drive the tracing algorithm). When the universe of attribute authorities is empty, $\mathbb{A}$ is the trivial access structure $\{\emptyset\} \in 2^{\emptyset}$, and there is a single tracing authority, in which case Definition 5.17 collapses to (a syntactic variant of) the EIPLBE primitive given in [Zha25].

6 Decentralized Multi-Authority Attribute-Based Traitor Tracing

6.1 Definition

We define our new multi-authority attribute-based traitor tracing primitive.

Definition 6.1 (MA-AB-TT). A decentralized multi-authority attribute-based traitor tracing (MA-AB-TT) scheme for monotone access structures consists of eight procedures whose syntax is given below. We denote by $\mathcal{M}$ the supported message space and by $\mathcal{AID}$, $\mathcal{TID}$ and $\mathcal{GID}$ the (disjoint) universes of attribute authorities, tracing authorities and users, respectively. Additionally, we assume that each authority controls only one attribute, and hence we can use the words "authority" and "attribute" interchangeably. This definition naturally generalizes to the situation in which each authority potentially controls an arbitrary (bounded or unbounded) number of attributes (see [LW11, RW15]).

- $\mathsf{GlobalSetup}(1^\lambda) \to \mathsf{gp}$: On input the (unary encoded) security parameter 1^λ, it outputs the global public parameters gp for the system. We assume that gp includes the descriptions of $\mathcal{AID}$, $\mathcal{TID}$ and $\mathcal{GID}$. Note that $\mathcal{AID}$, $\mathcal{TID}$ and $\mathcal{GID}$ are given by $\{0,1\}^\lambda$ in case there is no bound on the number of authorities or users.
- $\mathsf{TrAuthSetup}(\mathsf{gp}, \mathsf{tid}) \to (\mathsf{pk}_{\mathsf{tid}}, \mathsf{msk}_{\mathsf{tid}})$: On input the global parameters gp and a tracing authority $\mathsf{tid} \in \mathcal{TID}$, it outputs the authority's public/master secret key pair $(\mathsf{pk}_{\mathsf{tid}}, \mathsf{msk}_{\mathsf{tid}})$.
- $\mathsf{AttAuthSetup}(\mathsf{gp}, \mathsf{aid}) \to (\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}})$: On input the global parameters gp and an attribute authority $\mathsf{aid} \in \mathcal{AID}$, it outputs the authority's public/master secret key pair $(\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}})$.
- $\mathsf{TrKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{tid}}, \mathsf{gid}) \to \mathsf{sk}_{\mathsf{tid},\mathsf{gid}}$: On input the global parameters gp, the master secret key $\mathsf{msk}_{\mathsf{tid}}$ of a tracing authority $\mathsf{tid} \in \mathcal{TID}$, and a user's global identifier $\mathsf{gid} \in \mathcal{GID}$, it outputs a tracing secret key $\mathsf{sk}_{\mathsf{tid},\mathsf{gid}}$ for the user.
- $\mathsf{AttKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, \mathsf{gid}) \to \mathsf{sk}_{\mathsf{aid},\mathsf{gid}}$: On input the global parameters gp, the master secret key $\mathsf{msk}_{\mathsf{aid}}$ of an attribute authority $\mathsf{aid} \in \mathcal{AID}$, and a user's global identifier $\mathsf{gid} \in \mathcal{GID}$, it outputs an attribute secret key $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}}$ for the user.

- $\mathsf{Enc}(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu) \to \mathsf{ct}$: On input the global parameters gp, the public keys $\{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}$ for a set of tracing authorities $T \subseteq \mathcal{TID}$, a monotone access structure $\mathbb{T} \subseteq 2^{\mathcal{TID}}$ on T, the public keys $\{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}$ for a set of attribute authorities $A \subseteq \mathcal{AID}$, a monotone access structure $\mathbb{A} \subseteq 2^{\mathcal{AID}}$ on A, and a message $\mu \in \mathcal{M}$, it outputs a ciphertext ct.
- $\mathsf{Dec}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{tid,gid}}\}_{\mathsf{tid}\in T}, \{\mathsf{sk}_{\mathsf{aid,gid}}\}_{\mathsf{aid}\in A}, \mathsf{ct}) \to \mu' \vee \perp$: On input the global parameters gp, a set of secret keys $\{\mathsf{sk}_{\mathsf{tid,gid}}\}_{\mathsf{tid}\in T}$ issued by tracing authorities $\mathsf{tid} \in T$, a set of secret keys $\{\mathsf{sk}_{\mathsf{aid,gid}}\}_{\mathsf{aid}\in A}$ issued by attribute authorities $\mathsf{aid} \in A$, and a ciphertext ct, it outputs a message $\mu' \in \mathcal{M}$ or $\perp$ (a designated error symbol).
- $\mathsf{Trace}^{D}(\mathsf{gp}, 1^{\lceil 1/\epsilon \rceil}, \mathsf{tk}_{\mathsf{tid}^*}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_0, \mu_1) \to J$: On input the global parameters gp, a (unary encoded) error bound $\epsilon = \epsilon(\lambda)$, the tracing key $\mathsf{tk}_{\mathsf{tid}^*} \in \{\mathsf{pk}_{\mathsf{tid}^*}, \mathsf{msk}_{\mathsf{tid}^*}\}$ of a tracing authority $\mathsf{tid}^* \in \mathcal{TID}$, the public keys $\{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}$ for a set of tracing authorities $T \subseteq (\mathcal{TID} \setminus \mathsf{tid}^*)$, a monotone access structure $\mathbb{T} \subseteq 2^{\mathcal{TID}}$ on $T^* \coloneqq T \cup \{\mathsf{tid}^*\}$, the public keys $\{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}$ for a set of attribute authorities $A \subseteq \mathcal{AID}$, a monotone access structure $\mathbb{A} \subseteq 2^{\mathcal{AID}}$ on A, and a pair of messages $\mu_0, \mu_1 \in \mathcal{M}$, along with oracle access to a "decoder" algorithm D, it outputs a set $J \subseteq \mathcal{GID}$. In particular, the output $\emptyset$ indicates a failed tracing procedure. If $\mathsf{tk}_{\mathsf{tid}^*} = \mathsf{pk}_{\mathsf{tid}^*}$ (resp. $\mathsf{tk}_{\mathsf{tid}^*} = \mathsf{msk}_{\mathsf{tid}^*}$), the scheme is said to be publicly (resp. secretly) traceable.

Moreover, an MA-AB-TT scheme Π should satisfy the following properties.

- **Correctness:** We say Π is correct if for every $\lambda \in \mathbb{N}$, every message $\mu \in \mathcal{M}$, every $\mathsf{gid} \in \mathcal{GID}$, every $T \subseteq \mathcal{TID}$, every $A \subseteq \mathcal{AID}$, every monotone access structure $\mathbb{T} \subseteq 2^{\mathcal{TID}}$ on T, every monotone access structure $\mathbb{A} \subseteq 2^{\mathcal{AID}}$ on A, every $T' \in \mathbb{T}$, and every $A' \in \mathbb{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that

$$\Pr\left[\mu = \mu' \;\middle|\; \begin{array}{l} \mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda) \\ \forall \mathsf{tid} \in T \cup T',\ (\mathsf{pk}_{\mathsf{tid}}, \mathsf{msk}_{\mathsf{tid}}) \leftarrow \mathsf{TrAuthSetup}(\mathsf{gp}, \mathsf{tid}) \\ \forall \mathsf{tid} \in T',\ \mathsf{sk}_{\mathsf{tid,gid}} \leftarrow \mathsf{TrKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{tid}}, \mathsf{gid}) \\ \forall \mathsf{aid} \in A \cup A',\ (\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}}) \leftarrow \mathsf{AttAuthSetup}(\mathsf{gp}, \mathsf{aid}) \\ \forall \mathsf{aid} \in A',\ \mathsf{sk}_{\mathsf{aid,gid}} \leftarrow \mathsf{AttKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, \mathsf{gid}) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu) \\ \mu' \coloneqq \mathsf{Dec}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{tid,gid}}\}_{\mathsf{tid}\in T'}, \{\mathsf{sk}_{\mathsf{aid,gid}}\}_{\mathsf{aid}\in A'}, \mathsf{ct}) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

- **IND-CPA security:** We say that Π is IND-CPA secure if for all stateful PPT adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\Pi,\mathcal{A}}(\lambda) \coloneqq \left|\Pr\left[\mathsf{ExpCPA}^{1}_{\Pi,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{ExpCPA}^{0}_{\Pi,\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda),$$

  where the experiment $\mathsf{ExpCPA}^{b}_{\Pi,\mathcal{A}}$ is defined in Figure 8, the adversary has access to the oracles $O \coloneqq \{O\mathsf{AttAuthSetup}, O\mathsf{AttCorrupt}, O\mathsf{AttKeyGen}\}$ and $C'_{\mathsf{att}} \coloneqq A \setminus \mathcal{K}_{\mathsf{att}}$.

- **Traceability:** Let $\epsilon = \epsilon(\lambda)$ and $\delta = \delta(\lambda)$ be functions of the security parameter λ. We say that Π is (ϵ, δ)-traceable, if for all stateful PPT adversaries $\mathcal{A}$ and all $\lambda \in \mathbb{N}$, the following two conditions hold

$$\Pr[\mathsf{GoodTr} \mid \mathsf{HonAuth}] \geq \Pr[\mathsf{GoodDec} \mid \mathsf{HonAuth}] - \delta(\lambda) \qquad \text{and} \qquad \Pr[\mathsf{BadTr}] \leq \delta(\lambda)\,,$$

  where the events GoodDec, GoodTr and BadTr are defined as follows:

  - GoodDec occurs when

$$\Pr\left[D(\mathsf{ct}) = b \mid b \leftarrow\!\!\$\ \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_b)\right] \geq \frac{1}{2} + \epsilon(\lambda).$$

  - GoodTr holds when

$$\exists T' \subseteq H_{\mathsf{trace}},\ T' \in \mathbb{T} \wedge \bigcap_{\mathsf{tid}\in T'} J_{\mathsf{tid}} \neq \emptyset.$$

$\mathsf{ExpCPA}^{b}_{\Pi,\mathcal{A}}(1^\lambda)$

$\mathcal{K}_{\mathrm{att}}, \mathcal{Q}_{\mathrm{att}} \coloneqq [\cdot];\ C_{\mathrm{att}} \coloneqq \emptyset$
$\mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$
$(\{\mathsf{pk}_{\mathrm{tid}}\}_{\mathrm{tid}\in T}, \mathbb{T}, A, \{\mathsf{pk}_{\mathrm{aid}}\}_{\mathrm{aid}\in C'_{\mathrm{att}}}, \mathbb{A}, \mu_0, \mu_1) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{gp})$
$\forall \mathrm{aid} \in A \setminus C'_{\mathrm{att}}$, **parse** $(\mathsf{pk}_{\mathrm{aid}}, \mathsf{msk}_{\mathrm{aid}}) \coloneqq \mathcal{K}_{\mathrm{att}}[\mathrm{aid}]$
$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{gp}, \{\mathsf{pk}_{\mathrm{tid}}\}_{\mathrm{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathrm{aid}}\}_{\mathrm{aid}\in A}, \mathbb{A}, \mu_b)$
$b' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{ct})$
assert $\forall \mathrm{gid} \in \mathcal{GID}$, $(\mathcal{Q}_{\mathrm{att}}[\mathrm{gid}] \cup C_{\mathrm{att}} \cup C'_{\mathrm{att}}) \notin \mathbb{A}$
return b'

$\mathsf{ExpTrace}^{\epsilon}_{\Pi,\mathcal{A}}(1^\lambda)$

$\mathcal{K}_{\mathrm{trace}}, \mathcal{K}_{\mathrm{att}}, \mathcal{Q}_{\mathrm{trace}}, \mathcal{Q}_{\mathrm{att}} \coloneqq [\cdot];\ C_{\mathrm{trace}}, C_{\mathrm{att}} \coloneqq \emptyset$
$\mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$
$(D, T, \{\mathsf{pk}_{\mathrm{tid}}\}_{\mathrm{tid}\in C'_{\mathrm{trace}}}, \mathbb{T}, A, \{\mathsf{pk}_{\mathrm{aid}}\}_{\mathrm{aid}\in C'_{\mathrm{att}}}, \mathbb{A}, \mu_0, \mu_1) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{gp})$
$J_{\mathrm{trace}} \coloneqq \{\mathrm{gid} \in \mathcal{GID} : (\mathcal{Q}_{\mathrm{trace}}[\mathrm{gid}] \cup C_{\mathrm{trace}} \cup C'_{\mathrm{trace}}) \in \mathbb{T}\}$
$J_{\mathrm{att}} \coloneqq \{\mathrm{gid} \in \mathcal{GID} : (\mathcal{Q}_{\mathrm{att}}[\mathrm{gid}] \cup C_{\mathrm{att}} \cup C'_{\mathrm{att}}) \in \mathbb{A}\}$
$J \coloneqq J_{\mathrm{trace}} \cap J_{\mathrm{att}}$
$\forall \mathrm{tid} \in T \setminus C'_{\mathrm{trace}}$, **parse** $(\mathsf{pk}_{\mathrm{tid}}, \mathsf{msk}_{\mathrm{tid}}) \coloneqq \mathcal{K}_{\mathrm{trace}}[\mathrm{tid}]$
$\forall \mathrm{aid} \in A \setminus C'_{\mathrm{att}}$, **parse** $(\mathsf{pk}_{\mathrm{aid}}, \mathsf{msk}_{\mathrm{aid}}) \coloneqq \mathcal{K}_{\mathrm{att}}[\mathrm{aid}]$
$\forall \mathrm{tid}^* \in H_{\mathrm{trace}}$, $J_{\mathrm{tid}^*} \leftarrow \mathsf{Trace}^{D}(\mathsf{gp}, 1^{\lceil 1/\epsilon \rceil}, \mathsf{tk}_{\mathrm{tid}^*}, \{\mathsf{pk}_{\mathrm{tid}}\}_{\mathrm{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathrm{aid}}\}_{\mathrm{aid}\in A}, \mathbb{A}, \mu_0, \mu_1)$
return $(J, \{J_{\mathrm{tid}}\}_{\mathrm{tid}\in H_{\mathrm{trace}}}, \mathsf{gp}, \{\mathsf{pk}_{\mathrm{tid}}\}_{\mathrm{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathrm{aid}}\}_{\mathrm{aid}\in A}, \mathbb{A}, \mu_0, \mu_1)$

$\mathcal{O}\mathsf{AttAuthSetup}(\mathrm{aid})$

assert $\mathcal{K}_{\mathrm{att}}[\mathrm{aid}] = \bot$
$(\mathsf{pk}_{\mathrm{aid}}, \mathsf{msk}_{\mathrm{aid}}) \leftarrow \mathsf{AttAuthSetup}(\mathsf{gp}, \mathrm{aid})$
$\mathcal{K}_{\mathrm{att}}[\mathrm{aid}] \coloneqq (\mathsf{pk}_{\mathrm{aid}}, \mathsf{msk}_{\mathrm{aid}})$
return $\mathsf{pk}_{\mathrm{aid}}$

$\mathcal{O}\mathsf{AttCorrupt}(\mathrm{aid})$

assert $\mathcal{K}_{\mathrm{att}}[\mathrm{aid}] \neq \bot$
$C_{\mathrm{att}} \coloneqq C_{\mathrm{att}} \cup \{\mathrm{aid}\}$
return $\mathcal{K}_{\mathrm{att}}[\mathrm{aid}]$

$\mathcal{O}\mathsf{AttKeyGen}(\mathrm{aid}, \mathrm{gid})$

assert $\mathcal{K}_{\mathrm{att}}[\mathrm{aid}] \neq \bot$
if $\mathcal{Q}_{\mathrm{att}}[\mathrm{gid}] = \bot$ **then** $\mathcal{Q}_{\mathrm{att}}[\mathrm{gid}] \coloneqq \emptyset$
$\mathcal{Q}_{\mathrm{att}}[\mathrm{gid}] \coloneqq \mathcal{Q}_{\mathrm{att}}[\mathrm{gid}] \cup \{\mathrm{aid}\}$
parse $(\mathsf{pk}_{\mathrm{aid}}, \mathsf{msk}_{\mathrm{aid}}) \coloneqq \mathcal{K}_{\mathrm{att}}[\mathrm{aid}]$
$\mathsf{sk}_{\mathrm{aid,gid}} \leftarrow \mathsf{AttKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathrm{aid}}, \mathrm{gid})$
return $\mathsf{sk}_{\mathrm{aid,gid}}$

$\mathcal{O}\mathsf{TrAuthSetup}(\mathrm{tid})$

assert $\mathcal{K}_{\mathrm{trace}}[\mathrm{tid}] = \bot$
$(\mathsf{pk}_{\mathrm{tid}}, \mathsf{msk}_{\mathrm{tid}}) \leftarrow \mathsf{TrAuthSetup}(\mathsf{gp}, \mathrm{tid})$
$\mathcal{K}_{\mathrm{trace}}[\mathrm{tid}] \coloneqq (\mathsf{pk}_{\mathrm{tid}}, \mathsf{msk}_{\mathrm{tid}})$
return $\mathsf{pk}_{\mathrm{tid}}$

$\mathcal{O}\mathsf{TrCorrupt}(\mathrm{tid})$

assert $\mathcal{K}_{\mathrm{trace}}[\mathrm{tid}] \neq \bot$
$C_{\mathrm{trace}} \coloneqq C_{\mathrm{trace}} \cup \{\mathrm{tid}\}$
return $\mathcal{K}_{\mathrm{trace}}[\mathrm{tid}]$

$\mathcal{O}\mathsf{TrKeyGen}(\mathrm{tid}, \mathrm{gid})$

assert $\mathcal{K}_{\mathrm{trace}}[\mathrm{tid}] \neq \bot$
if $\mathcal{Q}_{\mathrm{trace}}[\mathrm{gid}] = \bot$ **then** $\mathcal{Q}_{\mathrm{trace}}[\mathrm{gid}] \coloneqq \emptyset$
$\mathcal{Q}_{\mathrm{trace}}[\mathrm{gid}] \coloneqq \mathcal{Q}_{\mathrm{trace}}[\mathrm{gid}] \cup \{\mathrm{tid}\}$
parse $(\mathsf{pk}_{\mathrm{tid}}, \mathsf{msk}_{\mathrm{tid}}) \coloneqq \mathcal{K}_{\mathrm{trace}}[\mathrm{tid}]$
$\mathsf{sk}_{\mathrm{tid,gid}} \leftarrow \mathsf{TrKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathrm{tid}}, \mathrm{gid})$
return $\mathsf{sk}_{\mathrm{tid,gid}}$

Figure 8: Experiments $\mathsf{ExpCPA}^{b}_{\Pi,\mathcal{A}}$ and $\mathsf{ExpTrace}^{\epsilon}_{\Pi,\mathcal{A}}$ for Definition 6.1

- BadTr holds when $\exists \mathrm{tid} \in H_{\mathrm{trace}},\ J_{\mathrm{tid}} \notin J$.
- HonAuth holds when,

$$\forall \mathrm{gid} \in \mathcal{GID}, \forall S \subseteq (\mathcal{Q}_{\mathrm{trace}}[\mathrm{gid}] \cup C_{\mathrm{trace}} \cup C'_{\mathrm{trace}}),\ S \in \mathbb{T} \implies S \setminus (C_{\mathrm{trace}} \cup C'_{\mathrm{trace}}) \in \mathbb{T}.$$

  In other words, whenever the tracing keys available for gid—i.e., those issued by honest authorities $\mathcal{Q}_{\mathrm{trace}}[\mathrm{gid}]$ together with all corrupted and adversarially-keyed authorities $C_{\mathrm{trace}} \cup C'_{\mathrm{trace}}$—contain an authorized set, then they already contain an authorized set of *honest* authorities. Equivalently, every authorized tracing coalition that the adversary can embed in the decoder for gid contains an authorized subset of honest tracing authorities, which is what lets those authorities jointly trace gid.

Here, we use $(J, \{J_{\mathrm{tid}}\}_{\mathrm{tid}\in H_{\mathrm{trace}}}, \mathbb{A}, \mu_0, \mu_1, \mathsf{gp}, \mathsf{pk}_{\mathrm{tid}}, \{\mathsf{pk}_{\mathrm{aid}}\}_{\mathrm{aid}\in A}) \leftarrow \mathsf{ExpTrace}^{\epsilon}_{\Pi,\mathcal{A}}(1^\lambda)$, where the experiment $\mathsf{ExpTrace}^{\epsilon}_{\Pi,\mathcal{A}}$ is defined in Figure 8, the adversary has access to the oracles $\mathcal{O} \coloneqq \{\mathcal{O}\mathsf{TrAuthSetup}, \mathcal{O}\mathsf{TrCorrupt}, \mathcal{O}\mathsf{TrKeyGen}, \mathcal{O}\mathsf{AttAuthSetup}, \mathcal{O}\mathsf{AttCorrupt}, \mathcal{O}\mathsf{AttKeyGen}\}$ and $C'_{\mathrm{trace}} \coloneqq T \setminus \mathcal{K}_{\mathrm{trace}}$, $C'_{\mathrm{att}} \coloneqq A \setminus \mathcal{K}_{\mathrm{att}}$, $H_{\mathrm{trace}} \coloneqq T \setminus (C_{\mathrm{trace}} \cup C'_{\mathrm{trace}})$. We say that Π is traceable if there exists a negligible function $\nu = \mathrm{negl}(\lambda)$ such that Π is $(1/p, \nu)$-traceable for every polynomial $p = p(\lambda)$.

Remark 6.2 (Weaker notions of security). We consider the following weaker variants of the security definitions defined above.

- **Security against static corruptions:** We say that Π is secure against static corruptions if the adversary $\mathcal{A}$ in the experiments $\mathsf{ExpCPA}^{b}_{\Pi,\mathcal{A}}$ and $\mathsf{ExpTrace}^{\epsilon}_{\Pi,\mathcal{A}}$ can query the oracles $\mathcal{O}\mathsf{AttCorrupt}(\mathsf{aid})$ (resp. $\mathcal{O}\mathsf{TrCorrupt}(\mathsf{tid})$) only before making the first query $\mathcal{O}\mathsf{AttKeyGen}(\mathsf{aid},\cdot)$ (resp. $\mathcal{O}\mathsf{TrKeyGen}(\mathsf{tid},\cdot)$) and before receiving the challenge ciphertext ct.[16]
- **α-risky traceability:** Let $\alpha = \alpha(\lambda)$ be a function of the security parameter λ. We say that Π is α-risky if for all stateful PPT adversaries $\mathcal{A}$ and inverse polynomials ϵ, there exists a negligible function $\mathsf{negl}(\cdot)$, such that $\Pr[\mathsf{BadTr}] \leq \mathsf{negl}(\lambda)$ and $\Pr[\mathsf{GoodTr} \mid \mathsf{HonAuth}] \geq \alpha \cdot \Pr[\mathsf{GoodDec} \mid \mathsf{HonAuth}] - \mathsf{negl}(\lambda)$.
- **ε-limited traceability:** We say that Π is ϵ-limited if there exists a constant $\epsilon \in (0, 1/2)$ such that for all stateful PPT adversaries $\mathcal{A}$, there exist a negligible function $\mathsf{negl}(\cdot)$ such that $\Pr[\mathsf{BadTr}] \leq \mathsf{negl}(\lambda)$ and $\Pr[\mathsf{GoodTr} \mid \mathsf{HonAuth}] \geq \Pr[\mathsf{GoodDec} \mid \mathsf{HonAuth}] - \mathsf{negl}(\lambda)$.

Intuitively, in a risky traitor tracing scheme a traitor is only accused with some small but noticeable probability. On the other hand, in a limited traitor tracing scheme a malicious user is accused only for very good decoders that succeed a constant fraction of the time. In the case of limited traitor tracing schemes, the constant ϵ is hardcoded in the algorithm Trace and we omit $1^{\lceil 1/\epsilon \rceil}$ as an input to Trace. An MA-AB-TT scheme that is α-risky and ϵ-limited is referred to as (α, ϵ)-risky limited MA-AB-TT.

6.2 Construction of Secret-Key MA-AB-TT with Embedded Identities

6.2.1 Construction of Risky and Limited MA-AB-TT

We now adapt the EIPLBE-to-risky-limited-TT compiler given in [Zha25] to the multi-authority attribute-based setting. Concretely, we show that an MA-EIPLBE scheme (Definition 5.17) that is simultaneously decoder-based message hiding, decoder-based predicate hiding, and weak decoder-based mode indistinguishable (Definition 5.10) yields a risky limited index-based embedded-identity MA-AB-TT scheme. As in [GKW19, Zha25], we work in the *index-based embedded-identity* regime, and instantiate the user identifier space $\mathcal{GID}$ of Definition 6.1 as

$$\mathcal{GID} \coloneqq \mathcal{GID}_{\kappa,\lambda} = [2^{\kappa}] \times \{0,1\}^{\lambda},$$

such that each $\mathsf{gid} = (j, \mathsf{id}) \in \mathcal{GID}$ carries both an *index* $j \in [2^{\kappa}]$, on which the tracing algorithm probes the EIPLBE threshold predicates, and an *embedded identity* $\mathsf{id} \in \{0,1\}^{\lambda}$, which the tracing algorithm recovers bit by bit once j has been localized.[17] As in the index-based convention of [GKW19, Zha25], we further assume that the adversary $\mathcal{A}$ in $\mathsf{ExpTrace}^{\epsilon}_{\Pi,\mathcal{A}}$ (Figure 8) only queries $\mathcal{O}\mathsf{TrKeyGen}$ and $\mathcal{O}\mathsf{AttKeyGen}$ on identifiers $\mathsf{gid} = (j, \mathsf{id})$ whose first components j are pairwise distinct. A generic compiler in the spirit of [GKW19] subsequently strips the explicit index from TrKeyGen and AttKeyGen, yielding an MA-AB-TT scheme with $\mathcal{GID} = \{0,1\}^{\lambda}$.

High-level idea. The high-level idea of the construction is the same as that of [Zha25, Construction 51]. Concretely, the MA-AB-TT key-generation and decryption algorithms forward to those of MA-EIPLBE on user identifiers $\mathsf{gid} = (j, \mathsf{id})$, encryption is the public mode encryption of MA-EIPLBE on the combined access structure, and the tracing algorithm runs the EncSK' algorithm on the EIPLBE predicates $f_{t,i} \in \mathcal{F}^{\mathrm{EIPLBE}}_{\kappa,\lambda}$ from Section 5.4, scanning the index direction $i = 0$ to localize the index j^* of an accused user and then querying $i \in \{1, \ldots, \lambda\}$ to recover the embedded identity bit by bit. The only multi-authority adaptations are: (i) the tracing master key $\mathsf{msk}_{\mathsf{tid}^*}$ of the designated tracing authority plays the role of the EIPLBE master secret key for the special authority id^*, while the remaining tracing and attribute public keys are concatenated into a single set of EIPLBE authorities, and (ii) the MA-AB-TT access structures $(\mathbb{T}, \mathbb{A})$ are folded into a single MA-EIPLBE access structure that simultaneously enforces the tracing authority policy $\mathbb{T}$ and the attribute authority policy $\mathbb{A}$.

[16] Note that security with static corruptions implies adaptive corruptions via a standard complexity leveraging argument that incurs a security loss of 2^{Q} where Q denotes the number of queries to $\mathcal{O}\mathsf{AttAuthSetup}$ and $\mathcal{O}\mathsf{TrAuthSetup}$.

[17] Pinning the MA-EIPLBE embedded-identity parameter ℓ of Definition 5.17 to $\ell \coloneqq \lambda$ here is purely a notational convenience that matches the default identifier space $\mathcal{GID} = \{0,1\}^{\lambda}$ of Definition 6.1. The construction and proof go through verbatim with λ replaced by any polynomial $\ell = \ell(\lambda)$ for which the targeted identity bit-length suffices.

Construction 6.3 (Risky limited MA-AB-TT from MA-EIPLBE). The construction is parameterized by (κ, λ) and uses an MA-EIPLBE scheme

$$\Pi' = (\mathsf{GlobalSetup}', \mathsf{AuthSetup}', \mathsf{KeyGen}', \mathsf{EncPK}', \mathsf{EncSK}', \mathsf{Dec}'),$$

with parameters (κ, λ) over the authority space $\mathcal{ID} \coloneqq \mathcal{TID} \cup \mathcal{AID}$ (Definition 5.17). For two monotone access structures $\mathbb{T} \subseteq 2^{\mathcal{TID}}$ on $T \subseteq \mathcal{TID}$ and $\mathbb{A} \subseteq 2^{\mathcal{AID}}$ on $A \subseteq \mathcal{AID}$, we write

$$\mathbb{T} \boxtimes \mathbb{A} \coloneqq \{T' \cup A' \subseteq T \cup A \mid T' \in \mathbb{T},\, A' \in \mathbb{A}\} \subseteq 2^{T \cup A},$$

for the conjunctive product access structure, which is satisfied by exactly those key sets that simultaneously contain an authorized tracing-side set and an authorized attribute-side set. The compiled index-based embedded-identity MA-AB-TT scheme Π is defined as follows:

- $\mathsf{GlobalSetup}(1^\lambda)$ outputs $\mathsf{gp} \leftarrow \mathsf{GlobalSetup}'(1^\lambda)$.
- $\mathsf{TrAuthSetup}(\mathsf{gp}, \mathsf{tid})$ outputs $(\mathsf{pk}_{\mathsf{tid}}, \mathsf{msk}_{\mathsf{tid}}) \leftarrow \mathsf{AuthSetup}'(\mathsf{gp}, \mathsf{tid})$ for $\mathsf{tid} \in \mathcal{TID}$.
- $\mathsf{AttAuthSetup}(\mathsf{gp}, \mathsf{aid})$ outputs $(\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}}) \leftarrow \mathsf{AuthSetup}'(\mathsf{gp}, \mathsf{aid})$ for $\mathsf{aid} \in \mathcal{AID}$.
- $\mathsf{TrKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{tid}}, \mathsf{gid})$ outputs $\mathsf{sk}_{\mathsf{tid},\mathsf{gid}} \leftarrow \mathsf{KeyGen}'(\mathsf{gp}, \mathsf{msk}_{\mathsf{tid}}, \mathsf{gid})$ for $\mathsf{gid} \in \mathcal{GID}_{\kappa,\lambda}$.
- $\mathsf{AttKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, \mathsf{gid})$ outputs $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} \leftarrow \mathsf{KeyGen}'(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, \mathsf{gid})$ for $\mathsf{gid} \in \mathcal{GID}_{\kappa,\lambda}$.
- $\mathsf{Enc}(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu)$ designates $\mathsf{id}^* \coloneqq \min(T \cup A)$ (the lexicographically smallest authority identifier in $T \cup A$) and outputs $\mathsf{ct} \leftarrow \mathsf{EncPK}'(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in(T\cup A)\setminus\{\mathsf{id}^*\}}, \pi = (\mathbf{1}, \mathbb{T} \boxtimes \mathbb{A}), \mu)$.
- $\mathsf{Dec}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{tid},\mathsf{gid}}\}_{\mathsf{tid}\in T'}, \{\mathsf{sk}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid}\in A'}, \mathsf{ct})$ outputs $\mu \leftarrow \mathsf{Dec}'(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{id},\mathsf{gid}}\}_{\mathsf{id}\in T'\cup A'}, \mathsf{ct})$.
- $\mathsf{Trace}^D(\mathsf{gp}, \mathsf{msk}_{\mathsf{tid}^*}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_0, \mu_1)$ proceeds as follows. Set $A^\star \coloneqq T \cup A$ and $\mathbb{A}^\star \coloneqq \mathbb{T} \boxtimes \mathbb{A}$, with the designated MA-EIPLBE authority taken as $\mathsf{id}^* \coloneqq \mathsf{tid}^* \in T^\star \coloneqq T \cup \{\mathsf{tid}^*\}$. Let $\alpha, \beta, \gamma \in (0, 1)$ be the constants of the weak decoder-based mode indistinguishability of Π' (Definition 5.10). α is hardcoded as the limitedness threshold (i.e., the Trace algorithm is invoked with $\epsilon \coloneqq \alpha$) and γ controls the riskiness of the resulting scheme (i.e., $\delta \coloneqq \gamma$ and is only used in the analysis of Proposition 6.7). On the other hand, β governs the per-pair sampling and gap thresholds defined below. The algorithm executes the four steps below.

  1. **Probe the EIPLBE predicates $f_{t,i}$.** For each $(t, i) \in [0; \kappa] \times [0; \lambda]$, define

  $$p_{t,i} \coloneqq 2 \cdot \Pr\left[D(\mathsf{ct}) = b \;\middle|\; \begin{array}{l} b \leftarrow\!\$\ \{0,1\} \\ \mathsf{ct} \leftarrow \mathsf{EncSK}'(\mathsf{gp}, \mathsf{msk}_{\mathsf{tid}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in A^\star\setminus\{\mathsf{tid}^*\}}, \pi = (f_{t,i}, \mathbb{A}^\star), \mu_b) \end{array}\right] - 1$$

  (with $f_{t,i} \in \mathcal{F}^{\mathrm{EIPLBE}}_{\kappa,\lambda}$ as in Section 5.4), and form an empirical estimate $\widetilde{p}_{t,i}$ by drawing $\Theta(\lambda(\kappa+1)^2/\beta^2)$ independent samples. The number of samples is chosen such that, by a Hoeffding bound and a union bound over the $(\kappa+1)(\lambda+1)$ probed pairs,

  $$\Pr\left[\exists\,(t,i)\colon |\widetilde{p}_{t,i} - p_{t,i}| \geq \beta/12(\kappa+1)\right] \leq 2^{-\lambda}.$$

  2. **Localize candidate indices.** Set

  $$\mathcal{T}^* \coloneqq \{t \in [\kappa]\colon |\widetilde{p}_{t,0} - \widetilde{p}_{t-1,0}| > \beta/2(\kappa+1)\}.$$

  3. **Recover embedded identities.** For each $t \in \mathcal{T}^*$, define $\mathsf{id}_t \in \{0,1\}^\lambda$ bit-by-bit by setting, for $i \in [\lambda]$,

  $$(\mathsf{id}_t)_i \coloneqq \begin{cases} 1 & \text{if } |\widetilde{p}_{t,i} - \widetilde{p}_{t,0}| < |\widetilde{p}_{t,i} - \widetilde{p}_{t-1,0}|, \\ 0 & \text{otherwise.} \end{cases}$$

  4. **Output the accused users.** Return $J_{\mathsf{tid}^*} \coloneqq \{(t, \mathsf{id}_t)\colon t \in \mathcal{T}^*\} \subseteq [0;\kappa] \times \{0,1\}^\lambda \subseteq \mathcal{GID}_{\kappa,\lambda}$.

Remark 6.4. The tracing-authority policy $\mathbb{T}$ only enters Construction 6.3 through the combined access structure $\mathbb{A}^\star = \mathbb{T} \boxtimes \mathbb{A}$, which is supplied both to the MA-AB-TT encryption algorithm Enc (e.g., when $\mathsf{ExpTrace}^{\epsilon}_{\Pi,\mathcal{A}}$

samples the GoodDec test ciphertexts) and to the EncSK′ probes that Trace uses internally to estimate the per-position decoder advantages $\widetilde{p}_{t,i}$. The tracing algorithm itself operates with respect to a single designated authority tid^*, matching the single id^* structure of MA-EIPLBE (Definition 5.17). When $\mathbb{T}$ is non-trivial, the experiment $\mathsf{ExpTrace}^{\epsilon}_{\Pi,\mathcal{A}}$ of Figure 8 invokes Trace once for each $\mathsf{tid}^* \in H_{\mathrm{trace}}$, and the resulting per-authority sets $\{J_{\mathsf{tid}^*}\}_{\mathsf{tid}^* \in H_{\mathrm{trace}}}$ are combined inside the GoodTr/BadTr events.

Proposition 6.5 (Correctness). *If Π' is correct, then Π as in Construction 6.3 is correct.*

Proof. Fix $\lambda \in \mathbb{N}$, $\mu \in \mathcal{M}$, $\mathsf{gid} = (j, \mathsf{id}) \in \mathcal{GID}_{\kappa,\lambda}$, $T \subseteq \mathcal{TID}$, $A \subseteq \mathcal{AID}$, monotone access structures $\mathbb{T}, \mathbb{A}$, $T' \in \mathbb{T}$, and $A' \in \mathbb{A}$. Then, $T' \cup A' \in \mathbb{T} \boxtimes \mathbb{A}$, so the key set $\{\mathsf{sk}_{\mathsf{id},\mathsf{gid}}\}_{\mathsf{id} \in T' \cup A'}$ is authorized under $\mathbb{T} \boxtimes \mathbb{A}$. By the public-mode correctness of Π', $\mathsf{Dec}'(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{id},\mathsf{gid}}\}_{\mathsf{id} \in T' \cup A'}, \mathsf{ct})$ recovers μ from $\mathsf{ct} \leftarrow \mathsf{EncPK}'(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id} \in T \cup A}, \pi = (\mathbf{1}, \mathbb{T} \boxtimes \mathbb{A}), \mu)$ with overwhelming probability, which is exactly the output of Dec on Π. □

Proposition 6.6 (IND-CPA security). *If Π' is (0-bounded) mode indistinguishable and (0-bounded) message hiding (Definition 5.1), and oblivious to id^* (Definition 5.4), then Π as in Construction 6.3 is IND-CPA secure.*

Proof. Consider a PPT adversary $\mathcal{A}$ playing $\mathsf{ExpCPA}^{b}_{\Pi,\mathcal{A}}$ (Figure 8), with output $(\{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid} \in T}, \mathbb{T}, A, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid} \in C'_{\mathrm{att}}}, \mathbb{A}, \mu_0, \mu_1)$. Write $A^\star := T \cup A$ and $\mathbb{A}^\star := \mathbb{T} \boxtimes \mathbb{A}$ as in Construction 6.3, such that, with the designated authority $\mathsf{id}^*_0 := \min(A^\star)$ of Construction 6.3, the challenge ciphertext on bit b is $\mathsf{ct} \leftarrow \mathsf{EncPK}'(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*_0}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id} \in A^\star \setminus \{\mathsf{id}^*_0\}}, \pi = (\mathbf{1}, \mathbb{A}^\star), \mu_b)$. Without loss of generality, fix an honest attribute authority $\mathsf{id}^* \in A \setminus (C_{\mathrm{att}} \cup C'_{\mathrm{att}})$ to play the role of the designated MA-EIPLBE authority. If no such id^* exists, then $A \subseteq C_{\mathrm{att}} \cup C'_{\mathrm{att}}$ and $\mathcal{A}$'s admissibility $(Q_{\mathrm{att}}[\mathsf{gid}] \cup C_{\mathrm{att}} \cup C'_{\mathrm{att}}) \notin \mathbb{A}$ degenerates to $A \notin \mathbb{A}$, which makes $\mathbb{A}^\star$ unsatisfiable on the attribute side and renders Π trivially IND-CPA secure. Since this challenge encrypts under the all-accepting predicate $\mathbf{1}$, obliviousness to id^* (Definition 5.4) ensures that re-designating the special authority from id^*_0 to the honest id^* leaves the ciphertext distribution unchanged, and hence, we may assume the challenge is $\mathsf{ct} \leftarrow \mathsf{EncPK}'(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id} \in A^\star \setminus \{\mathsf{id}^*\}}, \pi = (\mathbf{1}, \mathbb{A}^\star), \mu_b)$. Let $f := f_{2^\kappa,0} \in \mathcal{F}^{\mathrm{EIPLBE}}_{\kappa,\lambda}$ be the all-accepting predicate (i.e., $f(\mathsf{gid}) = 1$ for every $\mathsf{gid} \in \mathcal{GID}_{\kappa,\lambda}$).

Consider the following hybrid sequence:

$$
\begin{aligned}
H_0 &: \quad \mathsf{ct} \leftarrow \mathsf{EncPK}'(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id} \in A^\star \setminus \{\mathsf{id}^*\}}, \pi = (\mathbf{1}, \mathbb{A}^\star), \mu_0),\\
H_1 &: \quad \mathsf{ct} \leftarrow \mathsf{EncSK}'(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id} \in A^\star \setminus \{\mathsf{id}^*\}}, \pi = (f, \mathbb{A}^\star), \mu_0),\\
H_2 &: \quad \mathsf{ct} \leftarrow \mathsf{EncSK}'(\mathsf{gp}, \mathsf{msk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id} \in A^\star \setminus \{\mathsf{id}^*\}}, \pi = (f, \mathbb{A}^\star), \mu_1),\\
H_3 &: \quad \mathsf{ct} \leftarrow \mathsf{EncPK}'(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id} \in A^\star \setminus \{\mathsf{id}^*\}}, \pi = (\mathbf{1}, \mathbb{A}^\star), \mu_1),
\end{aligned}
$$

such that $H_0 = \mathsf{ExpCPA}^{0}_{\Pi,\mathcal{A}}$ and $H_3 = \mathsf{ExpCPA}^{1}_{\Pi,\mathcal{A}}$. In each transition, a (black-box and straight-line) reduction $\mathcal{A}'$ runs $\mathcal{A}$, with id^* committed as the special MA-EIPLBE authority. Since Π''s mode-indistinguishability game does not expose a $O\mathsf{Corrupt}$ oracle whereas its message-hiding game does (Definition 5.1), the routing of $\mathcal{A}$'s queries differs between the two transition types, which we describe below.

- $H_0 \approx_c H_1$ **and** $H_2 \approx_c H_3$ **by mode indistinguishability of Π'.** For these transitions, $\mathcal{A}'$ registers only id^* with Π' via $O\mathsf{AuthSetup}(\mathsf{id}^*)$ and generates all other attribute authorities locally, sampling $(\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}}) \leftarrow \mathsf{AuthSetup}(\mathsf{gp}, \mathsf{aid})$ itself and storing the result. Concretely, $\mathcal{A}$'s $O\mathsf{AttAuthSetup}(\mathsf{id}^*)$ is forwarded to Π''s $O\mathsf{AuthSetup}(\mathsf{id}^*)$, while $O\mathsf{AttAuthSetup}(\mathsf{aid})$ for $\mathsf{aid} \neq \mathsf{id}^*$ is answered locally. $O\mathsf{AttKeyGen}(\mathsf{id}^*, \mathsf{gid})$ is forwarded to Π''s $O\mathsf{KeyGen}(\mathsf{id}^*, \mathsf{gid})$, while $O\mathsf{AttKeyGen}(\mathsf{aid}, \mathsf{gid})$ for $\mathsf{aid} \neq \mathsf{id}^*$ is computed locally using the stored $\mathsf{msk}_{\mathsf{aid}}$. Moreover, $O\mathsf{AttCorrupt}(\mathsf{aid})$ is answered locally with the stored $(\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}})$ (note that $\mathcal{A}$ never corrupts id^* since by choice $\mathsf{id}^* \in A \setminus (C_{\mathrm{att}} \cup C'_{\mathrm{att}})$). Since the mode indistinguishability admissibility condition depends neither on C, C' nor f, this simulation is faithful. Hence $|\Pr[\mathcal{A}(H_0) = 1] - \Pr[\mathcal{A}(H_1) = 1]|$ and $|\Pr[\mathcal{A}(H_2) = 1] - \Pr[\mathcal{A}(H_3) = 1]|$ are both bounded by the mode indistinguishability advantage of $\mathcal{A}'$ against Π', which is negligible by assumption.
- $H_1 \approx_c H_2$ **by message hiding of Π'.** For this transition, since Π''s message hiding game exposes $O\mathsf{Corrupt}$, $\mathcal{A}'$ forwards each query to $O\mathsf{AttAuthSetup}$, $O\mathsf{AttCorrupt}$, $O\mathsf{AttKeyGen}$ verbatim to Π''s oracles $O\mathsf{AuthSetup}$, $O\mathsf{Corrupt}$, $O\mathsf{KeyGen}$. This routing induces the following correspondence between Π''s bookkeeping sets and $\mathcal{A}$'s MA-AB-TT side: $\mathcal{I}[\mathsf{gid}] = Q_{\mathrm{att}}[\mathsf{gid}] \cup C_{\mathrm{att}} \cup C'_{\mathrm{att}}$ (since only attribute-side $O\mathsf{KeyGen}$ queries are made), $C = C_{\mathrm{att}}$ (since only attribute-side $O\mathsf{Corrupt}$ queries are made), and $C' = T \cup C'_{\mathrm{att}}$ (the tracing authorities supplied by $\mathcal{A}$ together with the adversary supplied attribute authorities). Plugging these values along with the choice $f = \mathbf{1}$ into the admissibility condition of Π''s game $\mathsf{ExpMsg}^{q=0,b}_{\Pi',\mathcal{A}'}$, we

obtain that $(Q_{\text{att}}[\text{gid}] \cup C_{\text{att}} \cup C'_{\text{att}} \cup T) \notin \mathbb{A}^\star$ for all gid. Hence, any set lying in $\mathbb{A}^\star = \mathbb{T} \boxtimes \mathbb{A}$ would have to contain an attribute side authorized subset within $Q_{\text{att}}[\text{gid}] \cup C_{\text{att}} \cup C'_{\text{att}}$, contradicting $\mathcal{A}$'s admissibility condition $(Q_{\text{att}}[\text{gid}] \cup C_{\text{att}} \cup C'_{\text{att}}) \notin \mathbb{A}$. Hence, $|\Pr[\mathcal{A}(H_1) = 1] - \Pr[\mathcal{A}(H_2) = 1]|$ is bounded by the message hiding advantage of $\mathcal{A}'$ against Π', which is negligible by assumption.

Combining the above yields $|\Pr[\mathsf{ExpCPA}^0_{\Pi,\mathcal{A}} = 1] - \Pr[\mathsf{ExpCPA}^1_{\Pi,\mathcal{A}} = 1]| \le \mathsf{negl}(\lambda)$, establishing IND-CPA security of Π. □

Proposition 6.7 (Risky limited traceability (adapted from [Zha25, Theorem 49])). *If Π' is simultaneously decoder-based message hiding, decoder-based predicate hiding and weak decoder-based mode indistinguishable with constants $\alpha, \beta, \gamma \in (0, 1)$ (Definition 5.10), and oblivious to id^* (Definition 5.4), then Π as in Construction 6.3 is a (γ, α)-risky limited traceable.*

Proof (sketch). Fix a PPT adversary $\mathcal{A}$ playing the traceability experiment $\mathsf{ExpTrace}^{\alpha}_{\Pi,\mathcal{A}}$ of Figure 8 (with the limitedness threshold instantiated by the constant α). Let

$$\left(J, \{J_{\mathsf{tid}^*}\}_{\mathsf{tid}^* \in H_{\text{trace}}}, \mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid} \in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid} \in A}, \mathbb{A}, \mu_0, \mu_1\right) \leftarrow \mathsf{ExpTrace}^{\alpha}_{\Pi,\mathcal{A}}(1^\lambda),$$

let D be the decoder produced by $\mathcal{A}$, and fix an arbitrary $\mathsf{tid}^* \in H_{\text{trace}}$ with combined authority set $A^\star = T \cup A$ and combined policy $\mathbb{A}^\star = \mathbb{T} \boxtimes \mathbb{A}$, as in Construction 6.3. Write $J = J_{\text{trace}} \cap J_{\text{att}}$ for the set of registered users—those $\text{gid} = (j, \mathsf{id})$ for which both the queried tracing-side keys form an authorized set under $\mathbb{T}$ *and* the queried attribute-side keys form an authorized set under $\mathbb{A}$ (as defined in $\mathsf{ExpTrace}^{\alpha}_{\Pi,\mathcal{A}}$ of Figure 8). Let $\mathcal{T}$ be the set of indices $j \in [0; \kappa]$ that appear as the first component of some $\text{gid} = (j, \mathsf{id}) \in J$.

Per-tid^* bounds. The per-tid^* bounds reduce to the decoder-based security properties of MA-EIPLBE Π'—specifically, decoder-based predicate hiding (Definition 5.10) for the first bound below, and weak decoder-based mode indistinguishability (Definition 5.10) together with decoder-based message hiding (Definition 5.10) for the second—via the proof of [Zha25, Theorem 49], which applies verbatim to our setting with two notational adaptations: (i) the straight-line reductions used to invoke decoder-based predicate hiding, decoder-based message hiding and weak decoder-based mode indistinguishability of Π' forward each of $\mathcal{A}$'s $O\mathsf{TrAuthSetup}$, $O\mathsf{AttAuthSetup}$, $O\mathsf{TrKeyGen}$, $O\mathsf{AttKeyGen}$ queries to Π''s $O\mathsf{AuthSetup}$, $O\mathsf{KeyGen}$ oracles, committing to tid^* as the designated EIPLBE authority id^*, and (ii) the MA-EIPLBE access-structure parameter is set to $\mathbb{A}^\star = \mathbb{T} \boxtimes \mathbb{A}$ throughout, such that the event $\mathsf{Good}^{\mathsf{pk}}_{\alpha}$ of Definition 5.10 on EncPK' ciphertexts under $\mathbb{A}^\star$ coincides with the event GoodDec of Definition 6.1 on Enc ciphertexts of Π. This last identification uses obliviousness to id^* (Definition 5.4): both $\mathsf{Good}^{\mathsf{pk}}_{\alpha}$ and GoodDec are measured on public-mode (EncPK') encryptions under the all-accepting predicate $\mathbf{1}$ (recall Enc encrypts under $\pi = (\mathbf{1}, \mathbb{A}^\star)$), so re-designating the special authority from $\mathsf{id}^* = \min(T \cup A)$ (used by Enc) to tid^* (committed by the reduction) leaves the ciphertext distribution unchanged. Obliviousness is needed only at the all-accepting predicate $\mathbf{1}$, since the probes at the non-trivial predicates $f_{t,i}$ all keep the single designation tid^* fixed, varying only the predicate (predicate hiding) or the mode (weak mode indistinguishability). With these adaptations, the sampling-precision $\beta/12(\kappa + 1)$ and gap-threshold calibration $\beta/2(\kappa + 1)$ of Construction 6.3 (matching that of [Zha25, Construction 51]) yields the per-tid^* bounds

$$\Pr[J_{\mathsf{tid}^*} \not\subseteq J] \le \mathsf{negl}(\lambda) \qquad \text{and} \qquad \Pr[\emptyset \subsetneq J_{\mathsf{tid}^*} \subseteq J] \ge \gamma \cdot \Pr[\text{GoodDec} \mid \text{HonAuth}] - \mathsf{negl}(\lambda),$$

where the second bound applies to each honest tid^* on whose key the decoder relies, and its non-empty J_{tid^*} contains the registered traitor $\text{gid}^\dagger = (t^\star, \mathsf{id}^\dagger) \in J$ at the dominant-jump index $t^\star \in \mathcal{T}$ identified by the pigeonhole step of [Zha25, Theorem 48] (i.e., the common user whose keys make up the decoder's authorized coalition).

Multi-tid aggregation. The aggregation of the per-tid^* bounds into the multi-tid events BadTr and GoodTr | HonAuth of Definition 6.1 has no analog in [Zha25], since that work focuses only on single tracing authority schemes. Hence, we explain the multi-tid extension below.

For BadTr, the event rewrites as $\exists \mathsf{tid} \in H_{\text{trace}},\ J_{\mathsf{tid}} \not\subseteq J$. By Boole's inequality and the per-tid^* bound,

$$\Pr[\text{BadTr}] \le \sum_{\mathsf{tid} \in H_{\text{trace}}} \Pr[J_{\mathsf{tid}} \not\subseteq J] \le |H_{\text{trace}}| \cdot \mathsf{negl}(\lambda) = \mathsf{negl}(\lambda).$$

For GoodTr | HonAuth, recall that on GoodDec the decoder D decrypts the real ciphertext, and therefore, relies for a single registered traitor $\mathsf{gid}^\dagger = (t^\star, \mathsf{id}^\dagger) \in J$, on an authorized tracing coalition $S \in \mathbb{T}$ of keys for $\mathsf{gid}^\dagger$. Since D can obtain keys for $\mathsf{gid}^\dagger$ only from honest authorities that issued them ($Q_{\mathrm{trace}}[\mathsf{gid}^\dagger]$) or from corrupted and adversarially-keyed authorities ($C_{\mathrm{trace}} \cup C'_{\mathrm{trace}}$), we have $S \subseteq Q_{\mathrm{trace}}[\mathsf{gid}^\dagger] \cup C_{\mathrm{trace}} \cup C'_{\mathrm{trace}}$. By HonAuth, the honest sub-coalition $T' \coloneqq S \setminus (C_{\mathrm{trace}} \cup C'_{\mathrm{trace}})$ satisfies $T' \in \mathbb{T}$ and $T' \subseteq H_{\mathrm{trace}}$. Each $\mathsf{tid} \in T'$ issued $\mathsf{gid}^\dagger$ a key on which D relies, so the per-tid correct-trace bound gives $\mathsf{gid}^\dagger \in J_{\mathsf{tid}}$. Hence, $\bigcap_{\mathsf{tid} \in T'} J_{\mathsf{tid}} \ni \mathsf{gid}^\dagger \neq \emptyset$, which together with $T' \in \mathbb{T}$ and $T' \subseteq H_{\mathrm{trace}}$ establishes GoodTr. This yields $\Pr[\mathsf{GoodTr} \mid \mathsf{HonAuth}] \geq \gamma \cdot \Pr[\mathsf{GoodDec} \mid \mathsf{HonAuth}] - \mathsf{negl}(\lambda)$. Therefore, given that α is hardcoded in Trace, the combined bounds $\Pr[\mathsf{BadTr}] \leq \mathsf{negl}(\lambda)$ and $\Pr[\mathsf{GoodTr} \mid \mathsf{HonAuth}] \geq \gamma \cdot \Pr[\mathsf{GoodDec} \mid \mathsf{HonAuth}] - \mathsf{negl}(\lambda)$ are precisely the statement that Π is (γ, α)-risky limited. □

6.2.2 Eliminating Riskiness

We adapt the riskiness mitigation compiler of [Zha20, DDP+25] to the multi-authority attribute-based setting. Given an (α, ϵ)-risky limited MA-AB-TT scheme

$$\Pi' = (\mathsf{GlobalSetup}', \mathsf{AttAuthSetup}', \mathsf{TrAuthSetup}', \mathsf{TrKeyGen}', \mathsf{AttKeyGen}', \mathsf{Enc}', \mathsf{Dec}', \mathsf{Trace}'),$$

and a parameter $\beta \in (\alpha, 1]$, the compiler produces a (β, ϵ)-risky limited MA-AB-TT scheme Π. Setting $\beta = 1$ removes riskiness entirely, yielding a (non-risky) ϵ-limited MA-AB-TT scheme.

The high-level idea is the same as in [Zha20, DDP+25], i.e., we run $\tau = (\beta/\alpha) \cdot \omega(\log \lambda)$ parallel copies of Π'. Every authority—whether attribute or tracing—executes its setup procedure independently τ times, producing τ independent key pairs. The encryptor samples a fresh index $j \leftarrow\!\!\$\ [\tau]$ and encrypts only under the j-th instance. To trace a decoder D on behalf of a tracing authority tid^*, the tracing algorithm constructs, for each instance $j \in [\tau]$, a derived decoder D_j that emulates D on ciphertexts of the j-th instance, runs the underlying tracer on D_j, and outputs the union $J_{\mathsf{tid}^*} = \bigcup_{j \in [\tau]} J_{\mathsf{tid}^*, j}$ of the resulting traced sets. Since the experiment in Definition 6.1 combines the per-authority sets $\{J_{\mathsf{tid}^*}\}_{\mathsf{tid}^* \in H_{\mathrm{trace}}}$ by intersecting over an authorized set $T' \in \mathbb{T}$, and intersection is monotone under taking unions, a successful trace in a *single* good instance already suffices to make the compiled trace succeed, which is precisely what enables the amplification from α to β.

Construction 6.8 (Riskiness mitigation for MA-AB-TT). The construction is parameterized by $\tau = \tau(\lambda) \in \mathbb{N}$ and uses an (α, ϵ)-risky limited MA-AB-TT scheme Π' as a building block. The compiled scheme Π is defined as follows.

- $\mathsf{GlobalSetup}(1^\lambda)$ runs $\mathsf{gp}_j \leftarrow \mathsf{GlobalSetup}'(1^\lambda)$ for each $j \in [\tau]$ and outputs $\mathsf{gp} \coloneqq (\mathsf{gp}_j)_{j \in [\tau]}$.
- $\mathsf{AttAuthSetup}(\mathsf{gp}, \mathsf{aid})$ parses $\mathsf{gp} = (\mathsf{gp}_j)_{j \in [\tau]}$. For each $j \in [\tau]$, it runs

  $$(\mathsf{pk}_{\mathsf{aid},j}, \mathsf{msk}_{\mathsf{aid},j}) \leftarrow \mathsf{AttAuthSetup}'(\mathsf{gp}_j, \mathsf{aid}).$$

  It outputs $\mathsf{pk}_{\mathsf{aid}} \coloneqq (\mathsf{pk}_{\mathsf{aid},j})_{j \in [\tau]}$ and $\mathsf{msk}_{\mathsf{aid}} \coloneqq (\mathsf{msk}_{\mathsf{aid},j})_{j \in [\tau]}$.
- $\mathsf{TrAuthSetup}(\mathsf{gp}, \mathsf{tid})$ parses $\mathsf{gp} = (\mathsf{gp}_j)_{j \in [\tau]}$. For each $j \in [\tau]$, it runs

  $$(\mathsf{pk}_{\mathsf{tid},j}, \mathsf{msk}_{\mathsf{tid},j}) \leftarrow \mathsf{TrAuthSetup}'(\mathsf{gp}_j, \mathsf{tid}).$$

  It outputs $\mathsf{pk}_{\mathsf{tid}} \coloneqq (\mathsf{pk}_{\mathsf{tid},j})_{j \in [\tau]}$ and $\mathsf{msk}_{\mathsf{tid}} \coloneqq (\mathsf{msk}_{\mathsf{tid},j})_{j \in [\tau]}$.
- $\mathsf{TrKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{tid}}, \mathsf{gid})$ parses $\mathsf{gp} = (\mathsf{gp}_j)_{j \in [\tau]}$ and $\mathsf{msk}_{\mathsf{tid}} = (\mathsf{msk}_{\mathsf{tid},j})_{j \in [\tau]}$. For each $j \in [\tau]$, it runs

  $$\mathsf{sk}_{\mathsf{tid},\mathsf{gid},j} \leftarrow \mathsf{TrKeyGen}'(\mathsf{gp}_j, \mathsf{msk}_{\mathsf{tid},j}, \mathsf{gid}),$$

  and outputs $\mathsf{sk}_{\mathsf{tid},\mathsf{gid}} \coloneqq (\mathsf{sk}_{\mathsf{tid},\mathsf{gid},j})_{j \in [\tau]}$.
- $\mathsf{AttKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, \mathsf{gid})$ parses $\mathsf{gp} = (\mathsf{gp}_j)_{j \in [\tau]}$ and $\mathsf{msk}_{\mathsf{aid}} = (\mathsf{msk}_{\mathsf{aid},j})_{j \in [\tau]}$. For each $j \in [\tau]$, it runs $\mathsf{sk}_{\mathsf{aid},\mathsf{gid},j} \leftarrow \mathsf{AttKeyGen}'(\mathsf{gp}_j, \mathsf{msk}_{\mathsf{aid},j}, \mathsf{gid})$ and outputs $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} \coloneqq (\mathsf{sk}_{\mathsf{aid},\mathsf{gid},j})_{j \in [\tau]}$.
- $\mathsf{Enc}(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid} \in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid} \in A}, \mathbb{A}, \mu)$ parses $\mathsf{gp} = (\mathsf{gp}_j)_{j \in [\tau]}$ and, for each $\mathsf{tid} \in T$, $\mathsf{pk}_{\mathsf{tid}} = (\mathsf{pk}_{\mathsf{tid},j})_{j \in [\tau]}$,

and, for each $\mathsf{aid} \in A$, $\mathsf{pk}_{\mathsf{aid}} = (\mathsf{pk}_{\mathsf{aid},j})_{j\in[\tau]}$. It samples $j \leftarrow\!\!\$\ [\tau]$, runs

$$\mathsf{ct}_j \leftarrow \mathsf{Enc}'\big(\mathsf{gp}_j, \{\mathsf{pk}_{\mathsf{tid},j}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid},j}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu\big),$$

and outputs $\mathsf{ct} \coloneqq (j, \mathsf{ct}_j)$.

- $\mathsf{Dec}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{tid},\mathsf{gid}}\}_{\mathsf{tid}\in T}, \{\mathsf{sk}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid}\in A}, \mathsf{ct})$ parses $\mathsf{gp} = (\mathsf{gp}_j)_{j\in[\tau]}$, $\mathsf{ct} = (j, \mathsf{ct}_j)$, and, for each $\mathsf{tid} \in T$, $\mathsf{sk}_{\mathsf{tid},\mathsf{gid}} = (\mathsf{sk}_{\mathsf{tid},\mathsf{gid},j})_{j\in[\tau]}$, and, for each $\mathsf{aid} \in A$, $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} = (\mathsf{sk}_{\mathsf{aid},\mathsf{gid},j})_{j\in[\tau]}$. It outputs

$$\mathsf{Dec}'\big(\mathsf{gp}_j, \{\mathsf{sk}_{\mathsf{tid},\mathsf{gid},j}\}_{\mathsf{tid}\in T}, \{\mathsf{sk}_{\mathsf{aid},\mathsf{gid},j}\}_{\mathsf{aid}\in A}, \mathsf{ct}_j\big).$$

- $\mathsf{Trace}^D(\mathsf{gp}, \mathsf{tk}_{\mathsf{tid}^*}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_0, \mu_1)$ parses $\mathsf{gp} = (\mathsf{gp}_j)_{j\in[\tau]}$, $\mathsf{tk}_{\mathsf{tid}^*} = (\mathsf{tk}_{\mathsf{tid}^*,j})_{j\in[\tau]}$, and, for each $\mathsf{tid} \in T$ (resp. $\mathsf{aid} \in A$), $\mathsf{pk}_{\mathsf{tid}} = (\mathsf{pk}_{\mathsf{tid},j})_{j\in[\tau]}$ (resp. $\mathsf{pk}_{\mathsf{aid}} = (\mathsf{pk}_{\mathsf{aid},j})_{j\in[\tau]}$). For each $j \in [\tau]$, it defines the derived decoder

$$D_j(\mathsf{ct}') \coloneqq D\big((j, \mathsf{ct}')\big)$$

and runs

$$J_{\mathsf{tid}^*,j} \leftarrow \mathsf{Trace}'^{D_j}\big(\mathsf{gp}_j, \mathsf{tk}_{\mathsf{tid}^*,j}, \{\mathsf{pk}_{\mathsf{tid},j}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid},j}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_0, \mu_1\big).$$

It outputs $J_{\mathsf{tid}^*} \coloneqq \bigcup_{j\in[\tau]} J_{\mathsf{tid}^*,j}$.

Since Π' is ϵ-limited, the constant ϵ is hardcoded in Trace', and we accordingly omit $1^{\lceil 1/\epsilon \rceil}$ as an input to Trace.

Proposition 6.9 (Correctness). *If Π' is correct, then Π is correct.*

Proof. Fix $\lambda \in \mathbb{N}$, $\mu \in \mathcal{M}$, $\mathsf{gid} \in \mathcal{GID}$, $T \subseteq \mathcal{TID}$, $A \subseteq \mathcal{AID}$, monotone access structures $\mathbb{T} \subseteq 2^{\mathcal{TID}}$ on T and $\mathbb{A} \subseteq 2^{\mathcal{AID}}$ on A, and authorized sets $T' \in \mathbb{T}$, $A' \in \mathbb{A}$. Unrolling the algorithms of Π defined in Construction 6.8, an honest run yields, for every $j \in [\tau]$, a tuple

$$\big(\mathsf{gp}_j, \{(\mathsf{pk}_{\mathsf{tid},j}, \mathsf{msk}_{\mathsf{tid},j})\}_{\mathsf{tid}\in T\cup T'}, \{\mathsf{sk}_{\mathsf{tid},\mathsf{gid},j}\}_{\mathsf{tid}\in T'}, \{(\mathsf{pk}_{\mathsf{aid},j}, \mathsf{msk}_{\mathsf{aid},j})\}_{\mathsf{aid}\in A\cup A'}, \{\mathsf{sk}_{\mathsf{aid},\mathsf{gid},j}\}_{\mathsf{aid}\in A'}\big)$$

distributed as in the correctness experiment of Π'. Moreover, Enc samples $j^* \leftarrow\!\!\$\ [\tau]$ and outputs $\mathsf{ct} = (j^*, \mathsf{ct}_{j^*})$ with

$$\mathsf{ct}_{j^*} \leftarrow \mathsf{Enc}'\big(\mathsf{gp}_{j^*}, \{\mathsf{pk}_{\mathsf{tid},j^*}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid},j^*}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu\big).$$

Decryption then computes

$$\mu' = \mathsf{Dec}'\big(\mathsf{gp}_{j^*}, \{\mathsf{sk}_{\mathsf{tid},\mathsf{gid},j^*}\}_{\mathsf{tid}\in T'}, \{\mathsf{sk}_{\mathsf{aid},\mathsf{gid},j^*}\}_{\mathsf{aid}\in A'}, \mathsf{ct}_{j^*}\big).$$

The arguments to Dec′ are exactly the outputs of an honest run of the Π' correctness experiment with parameters $(\lambda, \mu, \mathsf{gid}, T, \mathbb{T}, A, \mathbb{A}, T', A')$. Since $T' \in \mathbb{T}$ and $A' \in \mathbb{A}$, correctness of Π' yields $\mu' = \mu$ with overwhelming probability. □

Proposition 6.10 (IND-CPA security). *If Π' is IND-CPA secure, then Π is IND-CPA secure. Concretely, for every PPT adversary $\mathcal{A}$ against Π, there exists a PPT adversary $\mathcal{A}'$ against Π' (with running time within a $\mathrm{poly}(\lambda, \tau)$ factor of $\mathcal{A}$'s) such that*

$$\mathsf{Adv}^{\text{ind-cpa}}_{\Pi,\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\text{ind-cpa}}_{\Pi',\mathcal{A}'}(\lambda).$$

Proof. We construct $\mathcal{A}'$, who plays $\mathsf{ExpCPA}^b_{\Pi',\mathcal{A}'}$ with its own challenger while emulating the role of the Π challenger towards $\mathcal{A}$. Throughout, $\mathcal{A}'$ embeds its Π' challenger into a single random instance $j^* \in [\tau]$ and simulates the remaining $\tau - 1$ instances itself.

Setup. On receiving gp' from its challenger, $\mathcal{A}'$ samples $j^* \leftarrow\!\!\$\ [\tau]$ and sets $\mathsf{gp}_{j^*} \coloneqq \mathsf{gp}'$. For every $j \in [\tau] \setminus \{j^*\}$, $\mathcal{A}'$ runs $\mathsf{gp}_j \leftarrow \mathsf{GlobalSetup}'(1^\lambda)$. It then invokes $\mathcal{A}$ on input $\mathsf{gp} \coloneqq (\mathsf{gp}_j)_{j\in[\tau]}$ and initializes empty dictionaries $\mathcal{K}_{\mathsf{att}}, \mathcal{Q}_{\mathsf{att}} \coloneqq [\cdot]$ and an empty corruption set $C_{\mathsf{att}} \coloneqq \emptyset$ exactly as in Figure 8.

Oracle simulation. $\mathcal{A}'$ answers $\mathcal{A}$'s oracle queries as follows.

- OAttAuthSetup(aid). For every $j \in [\tau] \setminus \{j^*\}$, $\mathcal{A}'$ runs $(\mathsf{pk}_{\mathsf{aid},j}, \mathsf{msk}_{\mathsf{aid},j}) \leftarrow \mathsf{AttAuthSetup}'(\mathsf{gp}_j, \mathsf{aid})$. For $j = j^*$, it forwards aid to its own OAttAuthSetup oracle to obtain $\mathsf{pk}_{\mathsf{aid},j^*}$ (the matching $\mathsf{msk}_{\mathsf{aid},j^*}$ remains with the Π' challenger). It records all collected components in $\mathcal{K}_{\mathsf{att}}[\mathsf{aid}]$ and returns $\mathsf{pk}_{\mathsf{aid}} \coloneqq (\mathsf{pk}_{\mathsf{aid},j})_{j\in[\tau]}$.
- OAttCorrupt(aid). $\mathcal{A}'$ forwards aid to its own OAttCorrupt oracle, which returns $(\mathsf{pk}_{\mathsf{aid},j^*}, \mathsf{msk}_{\mathsf{aid},j^*})$. Adding aid to C_{att}, $\mathcal{A}'$ returns $(\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}}) \coloneqq ((\mathsf{pk}_{\mathsf{aid},j})_{j\in[\tau]}, (\mathsf{msk}_{\mathsf{aid},j})_{j\in[\tau]})$.
- OAttKeyGen(aid, gid). For every $j \in [\tau] \setminus \{j^*\}$, $\mathcal{A}'$ runs $\mathsf{sk}_{\mathsf{aid},\mathsf{gid},j} \leftarrow \mathsf{AttKeyGen}'(\mathsf{gp}_j, \mathsf{msk}_{\mathsf{aid},j}, \mathsf{gid})$. For $j = j^*$, it forwards (aid, gid) to its own OAttKeyGen oracle to obtain $\mathsf{sk}_{\mathsf{aid},\mathsf{gid},j^*}$. Then, it sets $Q_{\mathsf{att}}[\mathsf{gid}] \coloneqq Q_{\mathsf{att}}[\mathsf{gid}] \cup \{\mathsf{aid}\}$, and returns $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} \coloneqq (\mathsf{sk}_{\mathsf{aid},\mathsf{gid},j})_{j\in[\tau]}$.

Recall that in ExpCPA the tracing authorities are entirely chosen by the adversary, so there are no tracing oracles to simulate.

Challenge. When $\mathcal{A}$ outputs the challenge tuple

$$(\{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, A, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in C'_{\mathsf{att}}}, \mathbb{A}, \mu_0, \mu_1)$$

with $C'_{\mathsf{att}} \coloneqq A \setminus \mathcal{K}_{\mathsf{att}}$, $\mathcal{A}'$ parses, for each $\mathsf{tid} \in T$, $\mathsf{pk}_{\mathsf{tid}} = (\mathsf{pk}_{\mathsf{tid},j})_{j\in[\tau]}$, and, for each $\mathsf{aid} \in C'_{\mathsf{att}}$, $\mathsf{pk}_{\mathsf{aid}} = (\mathsf{pk}_{\mathsf{aid},j})_{j\in[\tau]}$. It forwards

$$(\{\mathsf{pk}_{\mathsf{tid},j^*}\}_{\mathsf{tid}\in T}, \mathbb{T}, A, \{\mathsf{pk}_{\mathsf{aid},j^*}\}_{\mathsf{aid}\in C'_{\mathsf{att}}}, \mathbb{A}, \mu_0, \mu_1)$$

to its challenger and, on receiving the resulting challenge ciphertext ct', returns $\mathsf{ct} \coloneqq (j^*, \mathsf{ct}')$ to $\mathcal{A}$. After $\mathcal{A}$ outputs its guess b' (with continued oracle access simulated as above), $\mathcal{A}'$ outputs the same b'.

Analysis. The simulated view of $\mathcal{A}$ is distributed identically to its view in $\mathsf{ExpCPA}^b_{\Pi,\mathcal{A}}$:

- Each gp_j is an independently honest output of GlobalSetup′, so gp is distributed as in Π.GlobalSetup.
- For every authority aid, the components for $j \neq j^*$ are honestly generated by $\mathcal{A}'$, while those for $j = j^*$ are honestly generated by the Π' challenger; the joint distribution is therefore exactly that of Π.AttAuthSetup.
- The challenge ciphertext is $\mathsf{ct} = (j^*, \mathsf{ct}')$ where $j^* \leftarrow\!\!\$\ [\tau]$ and $\mathsf{ct}' \leftarrow \mathsf{Enc}'(\mathsf{gp}_{j^*}, \{\mathsf{pk}_{\mathsf{tid},j^*}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid},j^*}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_b)$, which matches the output distribution of Π.Enc exactly (since Π.Enc also samples its index uniformly from $[\tau]$).

Furthermore, $\mathcal{A}'$ tracks the same dictionaries $\mathcal{K}_{\mathsf{att}}, Q_{\mathsf{att}}$ and corruption set C_{att} for its own challenger as it does for $\mathcal{A}$, and the set C'_{att} submitted to its challenger coincides with the one submitted by $\mathcal{A}$. Therefore, $\mathcal{A}'$ is admissible in $\mathsf{ExpCPA}^b_{\Pi',\mathcal{A}'}$ exactly when $\mathcal{A}$ is admissible in $\mathsf{ExpCPA}^b_{\Pi,\mathcal{A}}$, and

$$\Pr\left[\mathsf{ExpCPA}^b_{\Pi,\mathcal{A}}(\lambda) = 1\right] = \Pr\left[\mathsf{ExpCPA}^b_{\Pi',\mathcal{A}'}(\lambda) = 1\right] \quad \text{for } b \in \{0,1\}.$$

Hence, the claimed advantages follow. □

Proposition 6.11 (Risky-limited traceability). *Let $\beta \in (\alpha, 1]$ and $\tau = (\beta/\alpha)\cdot\omega(\log\lambda)$. If Π' is an α-risky limited MA-AB-TT scheme, then Π is a β-risky limited MA-AB-TT scheme.*

Proof (sketch). Fix a PPT adversary $\mathcal{A}$ against Π, and let $\epsilon' \in (0, 1/2)$ denote the constant hardcoded in Π'.Trace′. Let D be the decoder output by $\mathcal{A}$ in $\mathsf{ExpTrace}_{\Pi,\mathcal{A}}$, and for each $j \in [\tau]$ define the per-instance decoder $D_j(\mathsf{ct}') \coloneqq D((j, \mathsf{ct}'))$. For every honest tracing authority $\mathsf{tid}^* \in H_{\mathsf{trace}}$, the compiled tracer runs

$$J_{\mathsf{tid}^*,j} \leftarrow \mathsf{Trace}'^{D_j}(\mathsf{gp}_j, \mathsf{tk}_{\mathsf{tid}^*,j}, \{\mathsf{pk}_{\mathsf{tid},j}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid},j}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_0, \mu_1) \quad \text{for each } j \in [\tau],$$

and outputs $J_{\mathsf{tid}^*} \coloneqq \bigcup_{j\in[\tau]} J_{\mathsf{tid}^*,j}$.

The compiled key generation oracles register each query $(\cdot, \mathsf{gid})$ in all τ instances simultaneously, and corruptions are likewise forwarded to all instances. Hence, the bookkeeping sets $Q_{\mathsf{trace}}, Q_{\mathsf{att}}, C_{\mathsf{trace}}, C_{\mathsf{att}}, C'_{\mathsf{trace}}, C'_{\mathsf{att}}$—and therefore the authorized set $J = J_{\mathsf{trace}} \cap J_{\mathsf{att}}$ and the honest set H_{trace}—are identical in $\mathsf{ExpTrace}_{\Pi,\mathcal{A}}$ and in the corresponding single-instance experiment of Π' restricted to any slot $j \in [\tau]$. In particular, the event HonAuth

(which is determined solely by the bookkeeping sets $Q_{\text{trace}}, C_{\text{trace}}, C'_{\text{trace}}$ and the policy $\mathbb{T}$) coincides across all these experiments. Here $J_{\text{tid}^*,j}$ is the set that Π' outputs in slot j for authority tid^*, such that $J_{\text{tid}^*} = \bigcup_{j\in[\tau]} J_{\text{tid}^*,j}$. We let GoodDec_j, GoodTr_j and BadTr_j denote the events GoodDec, GoodTr and BadTr of Π' in slot j with respect to the decoder D_j, where $\epsilon' \in (0, 1/2)$ is the limitedness constant hardcoded in $\Pi'.\text{Trace}'$:

$$\begin{aligned}
\text{GoodDec}_j &: \Pr\left[D_j(\text{ct}_j) = b \;\middle|\; b \leftarrow\!\$\ \{0,1\},\ \text{ct}_j \leftarrow \text{Enc}'(\text{gp}_j, \{\text{pk}_{\text{tid},j}\}_{\text{tid}\in T}, \mathbb{T}, \{\text{pk}_{\text{aid},j}\}_{\text{aid}\in A}, \mathbb{A}, \mu_b)\right] \geq \tfrac{1}{2} + \epsilon', \\
\text{GoodTr}_j &: \exists T' \subseteq H_{\text{trace}},\ T' \in \mathbb{T} \wedge \bigcap_{\text{tid}\in T'} J_{\text{tid},j} \neq \emptyset, \\
\text{BadTr}_j &: \exists \text{tid} \in H_{\text{trace}},\ J_{\text{tid},j} \not\subseteq J.
\end{aligned}$$

(Here $D_j(\text{ct}_j) = D((j, \text{ct}_j))$, so GoodDec_j is the advantage of D restricted to slot-j ciphertexts, measured against the limitedness threshold ϵ' of Π'.)

Bounding $\Pr[\text{BadTr}]$**.** The compiled event BadTr holds iff $\exists \text{tid}^* \in H_{\text{trace}},\ J_{\text{tid}^*} \not\subseteq J$. Since $J_{\text{tid}^*} = \bigcup_{j\in[\tau]} J_{\text{tid}^*,j}$, this implies $J_{\text{tid}^*,j} \not\subseteq J$ for some $j \in [\tau]$ and some $\text{tid}^* \in H_{\text{trace}}$, i.e., BadTr_j holds for some j. By the α-risky property of Π' applied in slot j (via the embedding described below),

$$\Pr[\text{BadTr}_j] \leq \text{negl}(\lambda) \quad \text{for every } j \in [\tau].$$

Since τ is polynomial in λ, a union bound yields $\Pr[\text{BadTr}] \leq \sum_{j\in[\tau]} \Pr[\text{BadTr}_j] \leq \tau \cdot \text{negl}(\lambda) = \text{negl}(\lambda)$.

Bounding $\Pr[\text{GoodTr}]$**.** The crucial observation is that a successful trace in a single instance already makes the compiled trace succeed. Indeed, suppose GoodTr_j holds for some slot j, witnessed by an authorized set $T' \subseteq H_{\text{trace}}$, $T' \in \mathbb{T}$ with $\bigcap_{\text{tid}\in T'} J_{\text{tid},j} \neq \emptyset$. Then, for the *same* T',

$$\bigcap_{\text{tid}\in T'} J_{\text{tid}} = \bigcap_{\text{tid}\in T'} \left(\bigcup_{j'\in[\tau]} J_{\text{tid},j'} \right) \supseteq \bigcap_{\text{tid}\in T'} J_{\text{tid},j} \neq \emptyset,$$

and hence, the compiled event GoodTr holds. Therefore, $\text{GoodTr} \supseteq \bigcup_{j\in[\tau]} \text{GoodTr}_j$, and it suffices to lower bound the probability that at least one instance traces successfully.

This reduces exactly to the situation analyzed in [Zha20, Theorem 10]: we have $\tau = (\beta/\alpha) \cdot \omega(\log\lambda)$ independent instances of an α-risky limited scheme, a single decoder D that decrypts slot-j ciphertexts through D_j, and—by the α-risky property of Π' in each slot—a per-instance tracing guarantee of the form $\Pr[\text{GoodTr}_j \mid \text{HonAuth}] \geq \alpha \cdot \Pr[\text{GoodDec}_j \mid \text{HonAuth}] - \text{negl}(\lambda)$. Because D depends on the keys of all τ instances, the events GoodTr_j are correlated. [Zha20, Theorem 10] circumvents this by reducing to a uniformly random target instance and coupling its tracing probability to the joint event via a careful calculation, which gives

$$\Pr\left[\bigcup_{j\in[\tau]} \text{GoodTr}_j \;\middle|\; \text{HonAuth}\right] \geq \beta \cdot \Pr[\text{GoodDec} \mid \text{HonAuth}] - \text{negl}(\lambda).$$

Combined with $\text{GoodTr} \supseteq \bigcup_{j\in[\tau]} \text{GoodTr}_j$, which gives $\Pr[\text{GoodTr} \mid \text{HonAuth}] \geq \Pr[\bigcup_{j\in[\tau]} \text{GoodTr}_j \mid \text{HonAuth}]$, this yields $\Pr[\text{GoodTr} \mid \text{HonAuth}] \geq \beta \cdot \Pr[\text{GoodDec} \mid \text{HonAuth}] - \text{negl}(\lambda)$, as required. Here we use that, because $\Pi.\text{Enc}$ samples a uniform slot $j \leftarrow\!\$\ [\tau]$ and sets $\text{ct} = (j, \text{ct}_j)$, the success probability of D against Π equals the slot-average $\frac{1}{\tau}\sum_{j\in[\tau]} \Pr[D_j(\text{ct}_j) = b]$ of its per-slot success probabilities. Hence the GoodDec event of Π—that this average is at least $\frac{1}{2} + \epsilon$—is identical to the good-decoder condition assumed in [Zha20, Theorem 10], whose compiled ciphertexts likewise target a uniformly random instance. Finally, since HonAuth is the same event in Π and in each slot experiment, the coupling goes through verbatim with every probability conditioned on HonAuth.

The multi-authority embedding. The only adaptation needed for the multi-authority setting is how the reduction $\mathcal{A}'$ embeds the single-instance α-risky Π' challenger into the random target instance $j^* \in [\tau]$ of Π. This is done exactly as in the IND-CPA reduction (Proposition 6.10): $\mathcal{A}'$ generates the $\tau - 1$ non-target instances itself by running $\Pi'.\text{GlobalSetup}', \Pi'.\text{AttAuthSetup}', \Pi'.\text{TrAuthSetup}', \Pi'.\text{TrKeyGen}'$ and $\Pi'.\text{AttKeyGen}'$, and forwards $\mathcal{A}$'s tracing- and attribute-key (and corruption) queries to its own oracles for the j^*-th component while running the corresponding Π' algorithms locally for the rest. Crucially, $\mathcal{A}'$ knows the master keys of every instance $j \neq j^*$, and hence, it can execute $\Pi'.\text{Trace}'$ locally on the derived decoders D_j needed by the coupling

argument's bookkeeping; the trace of the target instance j^* is the one captured by the Π' challenger's GoodTr event. With this embedding in place, the rest of the analysis from [Zha20, Theorem 10] applies without modification. □

6.2.3 Eliminating Limitedness

We adapt the limitedness mitigation compilers from [Zha20, DDP+25][18] to the MA-AB-TT setting. Given an ϵ-limited MA-AB-TT scheme

$$\Pi' = (\mathsf{GlobalSetup}', \mathsf{AttAuthSetup}', \mathsf{TrAuthSetup}', \mathsf{TrKeyGen}', \mathsf{AttKeyGen}', \mathsf{Enc}', \mathsf{Dec}', \mathsf{Trace}'),$$

with message space $\mathcal{M}' = \{0,1\}^\ell$, the compiler produces an MA-AB-TT scheme Π with message space $\mathcal{M} = \{0,1\}^z$ that satisfies the (plain) traceability property of Definition 6.1.

The high-level idea of our compiler is the same as in [Zha20, DDP+25]. That is, to encrypt a z-bit message $\mu \in \{0,1\}^z$, the compiled scheme generates $y \cdot z$ random "component" messages $\mu_{u,v} \in \mathcal{M}'$, for $u \in [y]$, $v \in [z]$, encrypts each independently using $\Pi'.\mathsf{Enc}'$, and masks μ with a key $k \in \{0,1\}^z$ obtained by applying a Goldreich-Levin hardcore predicate [GL89] to the column-wise XOR of the components. Intuitively, the components form a y-out-of-y secret sharing in each column, so a decoder for Π must successfully decrypt all y component ciphertexts in some column to recover any information about μ. Even an inverse-polynomial advantage against Π therefore implies a much higher per-component success rate against Π', which is what is needed to invoke Π''s limited tracing guarantee.

The compiler relies on the following standard result.

Theorem 6.12 (Goldreich-Levin Decoding [GL89]). *There exists a constant Γ and an oracle algorithm $\mathsf{GL}^D(\ell, \epsilon')$ running in time $\mathrm{poly}(\ell, \log(1/\epsilon'))$ and making $\mathrm{poly}(\ell, \log(1/\epsilon'))$ queries to D, such that the following holds. If there exists $x \in \{0,1\}^\ell$ such that*

$$\Pr[D(r) = x \cdot r \bmod 2 \mid r \leftarrow\!\$\ \{0,1\}^\ell] \geq \frac{1}{2} + \epsilon',$$

then $\Pr[\mathsf{GL}^D(\ell, \epsilon') = x] \geq \Gamma \cdot (\epsilon')^2$.

Construction 6.13 (Limitedness mitigation for MA-AB-TT). The construction is parameterized by $y = \omega(\log \lambda)$, $\ell = \omega(\lambda)$, $z = z(\lambda) \in \mathrm{poly}(\lambda)$, uses an ϵ-limited MA-AB-TT scheme Π' with message space $\mathcal{M}' = \{0,1\}^\ell$ as a building block, and a hash function $H\colon \{0,1\}^* \to \mathcal{M}' \times \mathcal{M}'$ that we model as a random oracle. The compiled scheme Π has message space $\mathcal{M} = \{0,1\}^z$ and is defined as follows.

- $\mathsf{GlobalSetup}$, $\mathsf{AttAuthSetup}$, $\mathsf{TrAuthSetup}$, $\mathsf{TrKeyGen}$, $\mathsf{AttKeyGen}$ are inherited from Π'.
- $\mathsf{Enc}(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu)$ does the following:
  1. For each $u \in [y]$, $v \in [z]$, samples $\mu_{u,v} \leftarrow\!\$\ \mathcal{M}'$ and computes
  $$\mathsf{ct}_{u,v} \leftarrow \mathsf{Enc}'\big(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_{u,v}\big).$$
  2. For each $v \in [z]$, sets $\mu_v \coloneqq \mu_{1,v} \oplus \cdots \oplus \mu_{y,v} \in \mathcal{M}'$.
  3. Samples $s \leftarrow\!\$\ \mathcal{M}'$ and sets $b_v \coloneqq \langle s, \mu_v \rangle \bmod 2$ for each $v \in [z]$, where $\langle \cdot, \cdot \rangle$ denotes the bitwise inner-product.
  4. Sets $k \coloneqq (b_1, \ldots, b_z) \in \{0,1\}^z$ and outputs $\mathsf{ct} \coloneqq (\mu \oplus k,\ s,\ \{\mathsf{ct}_{u,v}\}_{u\in[y], v\in[z]})$.
- $\mathsf{Dec}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{tid},\mathsf{gid}}\}_{\mathsf{tid}\in T}, \{\mathsf{sk}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid}\in A}, \mathsf{ct})$ parses $\mathsf{ct} = (\widehat{\mathsf{ct}}, s, \{\mathsf{ct}_{u,v}\})$. For each $u \in [y]$, $v \in [z]$, runs $\mu'_{u,v} \leftarrow \mathsf{Dec}'\big(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{tid},\mathsf{gid}}\}_{\mathsf{tid}\in T}, \{\mathsf{sk}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid}\in A}, \mathsf{ct}_{u,v}\big)$. For each $v \in [z]$, sets $\mu'_v \coloneqq \mu'_{1,v} \oplus \cdots \oplus \mu'_{y,v}$ and $b'_v \coloneqq \langle s, \mu'_v \rangle \bmod 2$. Outputs $\mu' \coloneqq \widehat{\mathsf{ct}} \oplus (b'_1, \ldots, b'_z)$.
- $\mathsf{Trace}^D(\mathsf{gp}, 1^{\lceil 1/\epsilon \rceil}, \mathsf{tk}_{\mathsf{tid}^*}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_0, \mu_1)$ sets $\epsilon' \coloneqq \epsilon/z$ and $\zeta \coloneqq (2yz^3/\Gamma\epsilon^3) \cdot \omega(\log \lambda)$, where Γ is the constant from Theorem 6.12. The algorithm proceeds in four steps.
  1. Constructs $D_1(s, \{c_u\}_{u\in[y]})$, taking as input $s \in \mathcal{M}'$ and y ciphertexts $c_1, \ldots, c_y$ of Π', as follows:

[18] We remark that in [Zha20] this property was called *threshold*, however, we follow the terminology of [DDP+25] and call this property *limitedness*.

- Samples $v \leftarrow\!\$\ [z]$, and for each $u \in [y]$, sets $c_{u,v} \coloneqq c_u$.
- For each $v' \in [z] \setminus \{v\}$ and $u \in [y]$, samples $\mu_{u,v'} \leftarrow\!\$\ \mathcal{M}'$ and runs

$$c_{u,v'} \leftarrow \mathsf{Enc}'\big(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_{u,v'}\big).$$

- For each $v' \in [z] \setminus \{v\}$, sets $\mu_{v'} \coloneqq \mu_{1,v'} \oplus \cdots \oplus \mu_{y,v'}$.
- For each $v' > v$, sets $b_{v'} \coloneqq \langle s, \mu_{v'}\rangle \bmod 2$. For each $v' \le v$, samples $b_{v'} \leftarrow\!\$\ \{0,1\}$. Sets $k \coloneqq (b_1, \ldots, b_z)$.
- Samples $\beta \leftarrow\!\$\ \{0,1\}$ and outputs $\beta \oplus b_v \oplus D\big(\mu_\beta \oplus k,\ s,\ \{c_{u,v'}\}_{u\in[y],\, v'\in[z]}\big)$.

2. Constructs $D_2(\{c_u\}_{u\in[y]}) \coloneqq \mathsf{GL}^{D_1(\cdot,\,(c_u)_{u\in[y]})}(\ell, \epsilon')$, where GL is the Goldreich-Levin extractor from Theorem 6.12.
3. Computes the masking pair $(\widehat{\mu}_0, \widehat{\mu}_1) \coloneqq H(D, \mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_0, \mu_1)$ and constructs $D_3(c)$ as follows. For $w \in [\zeta]$:
   - Samples $u \leftarrow\!\$\ [y]$ and sets $c_u \coloneqq c$.
   - For each $u' \in [y] \setminus \{u\}$, samples $\mu_{u'} \leftarrow\!\$\ \mathcal{M}'$ and computes

$$c_{u'} \leftarrow \mathsf{Enc}'\big(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_{u'}\big).$$

   - Runs $\mu' \leftarrow D_2(\{c_{u'}\}_{u'\in[y]})$ and sets $\mu^{(w)} \coloneqq \mu' \oplus \mu_1 \oplus \cdots \oplus \mu_{u-1} \oplus \mu_{u+1} \oplus \cdots \oplus \mu_y$.

   If $\mu^{(w)} = \widehat{\mu}_0$ for some $w \in [\zeta]$, outputs 0, and otherwise, outputs 1.
4. Returns $\mathsf{Trace}'^{D_3}\big(\mathsf{gp}, \mathsf{tk}_{\mathsf{tid}^*}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \widehat{\mu}_0, \widehat{\mu}_1\big)$.

Proposition 6.14 (Correctness). *If Π' is correct, then Π is correct.*

Proof. By correctness of Π', $\mu'_{u,v} = \mu_{u,v}$ for all $u \in [y]$, $v \in [z]$ with overwhelming probability. Hence, $\mu'_v = \mu_v$ and $b'_v = b_v$ for every v, and $\widehat{\mathsf{ct}} \oplus (b'_1, \ldots, b'_z) = (\mu \oplus k) \oplus k = \mu$. □

Proposition 6.15 (IND-CPA security). *If Π' is IND-CPA secure, then Π is IND-CPA secure. Concretely, for every PPT adversary $\mathcal{A}$ against Π, there exists a PPT adversary $\mathcal{A}'$ against Π' such that*

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\Pi,\mathcal{A}}(\lambda) \le 2z \cdot \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\Pi',\mathcal{A}'}(\lambda) + \mathsf{negl}(\lambda).$$

Proof. We first describe the reduction algorithm $\mathcal{A}'$. Upon receiving gp from the Π' challenger, $\mathcal{A}'$ invokes $\mathcal{A}$ on input gp and forwards $\mathcal{A}$'s oracle queries $\mathcal{O}\mathsf{AttAuthSetup}$, $\mathcal{O}\mathsf{AttCorrupt}$, $\mathcal{O}\mathsf{AttKeyGen}$ directly to its own oracles, and relays the responses to $\mathcal{A}$. Recall that in ExpCPA the tracing authorities are entirely chosen by the adversary, so there are no tracing oracles to simulate.

When $\mathcal{A}$ outputs the challenge tuple $(\{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, A, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in C'_{\mathsf{att}}}, \mathbb{A}, \mu_0, \mu_1)$, where $C'_{\mathsf{att}} \coloneqq A \setminus \mathcal{K}_{\mathsf{att}}$, $\mathcal{A}'$ samples $\widehat{\mu}_0, \widehat{\mu}_1 \leftarrow\!\$\ \mathcal{M}'$ and forwards $(\{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, A, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in C'_{\mathsf{att}}}, \mathbb{A}, \widehat{\mu}_0, \widehat{\mu}_1)$ to its own challenger. Let b denote the bit chosen by the Π' challenger, then $\mathcal{A}'$ receives back the challenge ciphertext $\mathsf{ct}^* \leftarrow \mathsf{Enc}'(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \widehat{\mu}_b)$.

To construct $\mathcal{A}$'s challenge ciphertext, $\mathcal{A}'$ proceeds as follows:

1. Samples $v^* \leftarrow\!\$\ [z]$, $u^* \leftarrow\!\$\ [y]$, and sets $\mathsf{ct}_{u^*,v^*} \coloneqq \mathsf{ct}^*$.
2. For each $(u,v) \ne (u^*, v^*)$, samples $\mu_{u,v} \leftarrow\!\$\ \mathcal{M}'$ and runs $\mathsf{ct}_{u,v} \leftarrow \mathsf{Enc}'(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_{u,v})$.
3. Samples $s \leftarrow\!\$\ \mathcal{M}'$ and $k^1 \leftarrow\!\$\ \{0,1\}^z$. If $\langle s, \widehat{\mu}_0\rangle = \langle s, \widehat{\mu}_1\rangle \bmod 2$, then aborts.
4. For each $v \ne v^*$, sets $\mu_v \coloneqq \mu_{1,v} \oplus \cdots \oplus \mu_{y,v}$ and $b_v \coloneqq \langle s, \mu_v\rangle \bmod 2$. Defines the key $k \in \{0,1\}^z$ by $k_i \coloneqq k^1_i$ for $i \le v^*$ and $k_i \coloneqq b_i$ for $i > v^*$. (Note that $k_{v^*} = k^1_{v^*}$ is uniform random and independent of $\mu_{u^*,v^*} = \widehat{\mu}_b$, which $\mathcal{A}'$ does not know.)
5. Samples $\beta \leftarrow\!\$\ \{0,1\}$ and sends $\mathsf{ct} \coloneqq (\mu_\beta \oplus k,\ s,\ \{\mathsf{ct}_{u,v}\}_{u,v})$ to $\mathcal{A}$.

When $\mathcal{A}$ outputs a guess β', $\mathcal{A}'$ computes

$$\widehat{b} \coloneqq \beta \oplus k^1_{v^*} \oplus \beta',$$

and outputs 0 if $\widehat{b} = \langle s, (\bigoplus_{u \neq u^*} \mu_{u,v^*}) \oplus \widehat{\mu}_0 \rangle \bmod 2$, and outputs 1 otherwise.

Next, we define a series of hybrids H_v for $v \in \{0, 1, \ldots, z\}$ as the modified IND-CPA game in which the key bits at positions $1, \ldots, v$ are sampled uniformly at random while the bits at positions $v+1, \ldots, z$ are computed honestly as $b_i = \langle s, \mu_i \rangle \bmod 2$. Then, we have that H_0 is the original IND-CPA game, and in H_z the entire key k is uniform, so the masked ciphertext $\mu_\beta \oplus k$ is independent of β, and $\mathcal{A}$ wins with probability exactly $1/2$. Let $p_v \coloneqq \Pr[\beta' = \beta \text{ in } H_v]$, i.e., the probability that $\mathcal{A}$ guesses correctly. Hence, we have that

$$p_0 = \frac{1}{2} + \frac{1}{2}\mathsf{Adv}^{\text{ind-cpa}}_{\Pi,\mathcal{A}}(\lambda), \qquad p_z = \frac{1}{2},$$

so $\eta \coloneqq p_0 - p_z = \frac{1}{2}\mathsf{Adv}^{\text{ind-cpa}}_{\Pi,\mathcal{A}}(\lambda)$.

Let H'_v denote H_v conditioned on the (random) bit at position v differing from $b_v = \langle s, \mu_v \rangle$. Since the random bit equals b_v with probability $1/2$ and conditioning on this event yields exactly H_{v-1}, we have $p_v = \frac{1}{2}p_{v-1} + \frac{1}{2}\Pr[\beta' = \beta \text{ in } H'_v]$, giving

$$\Pr[\beta' = \beta \text{ in } H'_v] = 2p_v - p_{v-1}.$$

Now, let E denote the abort event, i.e., $\langle s, \widehat{\mu}_0 \rangle = \langle s, \widehat{\mu}_1 \rangle \bmod 2$. For uniform $\widehat{\mu}_0, \widehat{\mu}_1, s \leftarrow\!\!\$\ \{0,1\}^\ell$ with $\ell = \omega(\lambda)$, $\Pr[E] = \frac{1}{2} + \mathsf{negl}(\lambda)$, and hence, $\Pr[\neg E] = \frac{1}{2} - \mathsf{negl}(\lambda)$.

Conditioned on $\neg E$, $v^* = v$ and the challenger's bit b, the correct hardcore bit at position v is

$$b_v^{(b)} \coloneqq \langle s, (\bigoplus_{u \neq u^*} \mu_{u,v}) \oplus \widehat{\mu}_b \rangle \bmod 2,$$

and $\neg E$ ensures $b_v^{(0)} \neq b_v^{(1)}$. Since k_v^1 is uniform on $\{0,1\}$, conditioned on $k_v^1 = b_v^{(b)}$ (probability $1/2$) the bit at position v in $\mathcal{A}'$'s simulated ciphertext equals the correct $b_v^{(b)}$, making the simulation identical to H_{v-1}. Conversely, conditioned on $k_v^1 \neq b_v^{(b)}$ (probability $1/2$) the simulation is identical to H'_v. Hence

$$\Pr[\beta' = \beta \mid \neg E, v^* = v, k_v^1 = b_v^{(b)}] = p_{v-1},$$
$$\Pr[\beta' = \beta \mid \neg E, v^* = v, k_v^1 \neq b_v^{(b)}] = 2p_v - p_{v-1}.$$

Now consider the event $\{\mathcal{A}' \text{ outputs } 0\}$ for fixed $b \in \{0,1\}$ and $v^* = v$, conditioned on $\neg E$. Recall that $\widehat{b} = \beta \oplus k_v^1 \oplus \beta'$, so $\widehat{b} = b_v^{(0)}$ iff $\beta' = \beta \oplus k_v^1 \oplus b_v^{(0)}$. Distinguishing the two sub-cases:

- $k_v^1 = b_v^{(b)}$: the simulation is of hybrid H_{v-1}, and hence, $\beta' = \beta$ with probability p_{v-1}. We then have $\widehat{b} = k_v^1 = b_v^{(b)}$, which equals $b_v^{(0)}$ iff $b = 0$. Therefore, $\Pr[\widehat{b} = b_v^{(0)} \mid b = 0, k_v^1 = b_v^{(0)}] = p_{v-1}$ and $\Pr[\widehat{b} = b_v^{(0)} \mid b = 1, k_v^1 = b_v^{(1)}] = 1 - p_{v-1}$.
- $k_v^1 \neq b_v^{(b)}$: the simulation is of hybrid H'_v, and hence, $\beta' = \beta$ with probability $2p_v - p_{v-1}$. Now $\widehat{b} = k_v^1 = b_v^{(1-b)}$ when $\beta' = \beta$, which equals $b_v^{(0)}$ iff $b = 1$. Therefore, $\Pr[\widehat{b} = b_v^{(0)} \mid b = 0, k_v^1 = b_v^{(1)}] = 1 - (2p_v - p_{v-1})$ and $\Pr[\widehat{b} = b_v^{(0)} \mid b = 1, k_v^1 = b_v^{(0)}] = 2p_v - p_{v-1}$.

Averaging over k_v^1 uniform on $\{0,1\}$ (equivalently, on $\{b_v^{(0)}, b_v^{(1)}\}$), we obtain

$$\Pr[\mathcal{A}' \text{ outputs } 0 \mid b = 0, \neg E, v^* = v] = \frac{1}{2}p_{v-1} + \frac{1}{2}(1 - (2p_v - p_{v-1})) = \frac{1}{2} + (p_{v-1} - p_v),$$
$$\Pr[\mathcal{A}' \text{ outputs } 0 \mid b = 1, \neg E, v^* = v] = \frac{1}{2}(1 - p_{v-1}) + \frac{1}{2}(2p_v - p_{v-1}) = \frac{1}{2} - (p_{v-1} - p_v).$$

On the other hand, when E occurs, $\mathcal{A}'$ outputs a uniformly random bit, contributing exactly 0 to its advantage.

Hence, we obtain

$$\begin{aligned}\Pr[\mathcal{A}' \text{ wins}] - \frac{1}{2} &= \frac{1}{2}\big(\Pr[\mathcal{A}' \text{ outputs } 0 \mid b = 0] - \Pr[\mathcal{A}' \text{ outputs } 0 \mid b = 1]\big) \\ &= \frac{1}{2} \cdot \Pr[\neg E] \cdot \sum_{v=1}^{z} \frac{1}{z}\big(\Pr[\mathcal{A}' \text{ outputs } 0 \mid b = 0, \neg E, v^* = v] \\ &\qquad - \Pr[\mathcal{A}' \text{ outputs } 0 \mid b = 1, \neg E, v^* = v]\big) \\ &= \frac{1}{2} \cdot \left(\frac{1}{2} - \mathsf{negl}(\lambda)\right) \cdot \sum_{v=1}^{z} \frac{1}{z} \cdot 2(p_{v-1} - p_v) \\ &= \frac{p_0 - p_z}{2z} - \mathsf{negl}(\lambda) = \frac{\eta}{2z} - \mathsf{negl}(\lambda).\end{aligned}$$

Since $\mathsf{Adv}^{\text{ind-cpa}}_{\Pi',\mathcal{A}'}(\lambda) = 2(\Pr[\mathcal{A}' \text{ wins}] - \frac{1}{2})$ and $\eta = \frac{1}{2}\mathsf{Adv}^{\text{ind-cpa}}_{\Pi,\mathcal{A}}(\lambda)$, this rearranges to

$$\mathsf{Adv}^{\text{ind-cpa}}_{\Pi,\mathcal{A}}(\lambda) \le 2z \cdot \mathsf{Adv}^{\text{ind-cpa}}_{\Pi',\mathcal{A}'}(\lambda) + \mathsf{negl}(\lambda),$$

as claimed. □

Proposition 6.16 (Traceability). *If Π' is an ϵ-limited MA-AB-TT scheme, then Π is a (non-limited) traceable MA-AB-TT scheme in the random oracle model.*

Proof (sketch). We outline the reduction from Π's traceability to the ϵ-limited property of Π', highlighting the multi-authority-specific points. The central Goldreich-Levin extraction and hybrid argument over the z columns are identical to those given in the proof of [Zha20, Theorem 9], to which we refer the reader for the full analysis.

Canonical derived decoder. The compiler is a single-instance transformation: the tracing layer (GlobalSetup, AttAuthSetup, TrAuthSetup, TrKeyGen, AttKeyGen) is inherited verbatim from Π', and Π.Trace merely replaces the decoder D by the derived decoder D_3 before delegating to Π'.Trace′. The experiment $\mathsf{ExpTrace}^{\epsilon}_{\Pi,\mathcal{A}}$ invokes Trace once for each honest tracing authority $\mathsf{tid}^* \in H_{\text{trace}}$, and *a priori* each invocation could build a different D_3 from its own fresh Goldreich-Levin randomness. Feeding Π' a different decoder per authority would make the per-authority traced sets $\{J_{\mathsf{tid}^*}\}$ independent, breaking the intersection clause $\bigcap_{\mathsf{tid}\in T'} J_{\mathsf{tid}} \neq \emptyset$ of GoodTr for conjunctive policies $\mathbb{T}$. We avoid this without assuming any coordination among authorities: the only construction-time randomness of D_3 is the masking pair $(\widehat{\mu}_0, \widehat{\mu}_1)$, and the construction derives it deterministically from the random oracle as

$$(\widehat{\mu}_0, \widehat{\mu}_1) = H(D, \mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_0, \mu_1).$$

Since the tracing key $\mathsf{tk}_{\mathsf{tid}^*}$ is *not* an input to H, every honest authority—tracing independently—constructs the very same decoder D_3. Thus, Π' is fed a single, canonical decoder across all authorities, exactly as its ϵ-limited guarantee requires. (The per-query randomness internal to D_3 may remain fresh; it is part of the single decoder that Π'.Trace′ queries, just as in Π''s own experiment.)

Reduction. Let $\mathcal{A}$ be a PPT adversary in $\mathsf{ExpTrace}^{\epsilon}_{\Pi,\mathcal{A}}$ (Figure 8). We construct $\mathcal{A}'$ in $\mathsf{ExpTrace}^{\epsilon}_{\Pi',\mathcal{A}'}$ as follows:

- $\mathcal{A}'$ forwards gp to $\mathcal{A}$ and relays *all* of $\mathcal{A}$'s oracle queries—OTrAuthSetup, OTrCorrupt, OTrKeyGen, OAttAuthSetup, OAttCorrupt, OAttKeyGen—verbatim to its own oracles, returning the responses to $\mathcal{A}$. Hence, we have that the bookkeeping sets $Q_{\text{trace}}, Q_{\text{att}}, C_{\text{trace}}, C_{\text{att}}, C'_{\text{trace}}, C'_{\text{att}}$ coincide. Consequently, the authorized set $J = J_{\text{trace}} \cap J_{\text{att}}$, the honest set H_{trace}, and the event HonAuth (which is determined solely by the bookkeeping sets $Q_{\text{trace}}, C_{\text{trace}}, C'_{\text{trace}}$ and the policy $\mathbb{T}$) are identical in $\mathsf{ExpTrace}_{\Pi,\mathcal{A}}$ and $\mathsf{ExpTrace}_{\Pi',\mathcal{A}'}$. The random oracle H is introduced by the compiler and is not part of Π', and hence, $\mathcal{A}'$ simulates it for $\mathcal{A}$ by lazy sampling: it answers each fresh query with a uniform value in $\mathcal{M}' \times \mathcal{M}'$ and replays cached answers on repeats, using the same table to compute the masking pair when it constructs D_3 below.
- When $\mathcal{A}$ outputs $(D, T, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in C'_{\text{trace}}}, \mathbb{T}, A, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in C'_{\text{att}}}, \mathbb{A}, \mu_0, \mu_1)$, $\mathcal{A}'$ computes $(\widehat{\mu}_0, \widehat{\mu}_1) = H(D, \mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_0, \mu_1)$, constructs the derived decoder D_3 (along with D_1, D_2) exactly as in Construction 6.13, and outputs $(D_3, T, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in C'_{\text{trace}}}, \mathbb{T}, A, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in C'_{\text{att}}}, \mathbb{A}, \widehat{\mu}_0, \widehat{\mu}_1)$ to its challenger.

For each $\mathsf{tid}^* \in H_{\mathsf{trace}}$, the Π' challenger computes $J_{\mathsf{tid}^*} \leftarrow \mathsf{Trace}'^{D_3}(\mathsf{gp}, \mathsf{tk}_{\mathsf{tid}^*}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \widehat{\mu}_0, \widehat{\mu}_1)$. Because the masking pair—and hence the description of D_3—is deterministically derived via $H(D, \mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_0, \mu_1)$ common to all authorities, this is exactly the collection $\{J_{\mathsf{tid}^*}\}_{\mathsf{tid}^*\in H_{\mathsf{trace}}}$ that $\Pi.\mathsf{Trace}$ produces in $\mathcal{A}$'s experiment, since $\Pi.\mathsf{Trace}$ delegates to Trace'^{D_3} with the same canonical D_3. Therefore, the per-authority traced sets agree pointwise, and—together with the fact that the combine ($\bigcap$ over authorized $T' \in \mathbb{T}$), the authorized set J, H_{trace}, and HonAuth all coincide—we obtain

$$\mathsf{BadTr}_{\mathcal{A}} \iff \mathsf{BadTr}_{\mathcal{A}'}, \quad \mathsf{GoodTr}_{\mathcal{A}} \iff \mathsf{GoodTr}_{\mathcal{A}'}, \quad \text{and} \quad \mathsf{HonAuth}_{\mathcal{A}} \iff \mathsf{HonAuth}_{\mathcal{A}'}.$$

Decoder advantage. By the Goldreich-Levin extraction argument given in the proof of [Zha20, Theorem 9], with the underlying encryption replaced with $\mathsf{Enc}'(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \cdot)$, the event $\mathsf{GoodDec}_{\mathcal{A}}$ (i.e., D has advantage $\geq \epsilon$ against Π) implies $\mathsf{GoodDec}_{\mathcal{A}'}$ (i.e., D_3 has constant distinguishing advantage against Π' on messages $\widehat{\mu}_0$ and $\widehat{\mu}_1$) up to negligible error. The only step that uses the distribution of the masking pair is the false-positive bound: D_3 errs on an $\mathsf{Enc}'(\widehat{\mu}_1)$ ciphertext only if $\widehat{\mu}_1 \in \{\mu^{(w)}\}_{w\in[\zeta]}$. In [Zha20, Theorem 9] the pair is sampled freshly after D is fixed, giving probability at most $\zeta/2^{\ell}$. Here $(\widehat{\mu}_0, \widehat{\mu}_1)$ is sampled via the random oracle H instead, though the candidate list $\{\mu^{(w)}\}$ is computed without reference to the masking pair, and hence, $\widehat{\mu}_1$ is uniform in $\mathcal{M}'$ and independent of the list, again giving probability at most $\zeta/2^{\ell}$ for each fixed D. A union bound over the polynomially many random-oracle queries of $\mathcal{A}$ keeps this negligible even though $\mathcal{A}$ may choose D adaptively after querying H.The hybrid bookkeeping is unaffected by the multi-authority structure because $\{\mathsf{pk}_{\mathsf{tid}}\}$, $\mathbb{T}$, $\{\mathsf{pk}_{\mathsf{aid}}\}$ and $\mathbb{A}$ enter only as input parameters to Enc' when D_1 and D_3 build the test ciphertexts; they do not interact with the Goldreich-Levin extraction itself. Since HonAuth coincides across the two experiments, this implication holds verbatim after conditioning on HonAuth, so $\Pr[\mathsf{GoodDec}_{\mathcal{A}'} \mid \mathsf{HonAuth}] \geq \Pr[\mathsf{GoodDec}_{\mathcal{A}} \mid \mathsf{HonAuth}] - \mathsf{negl}(\lambda)$. By the ϵ-limited property of Π' we have that

$$\Pr[\mathsf{GoodTr}_{\mathcal{A}'} \mid \mathsf{HonAuth}] \geq \Pr[\mathsf{GoodDec}_{\mathcal{A}'} \mid \mathsf{HonAuth}] - \mathsf{negl}(\lambda) \quad \text{and} \quad \Pr[\mathsf{BadTr}_{\mathcal{A}'}] \leq \mathsf{negl}(\lambda).$$

Translating the events back to $\mathcal{A}$'s experiment gives $\Pr[\mathsf{GoodTr}_{\mathcal{A}} \mid \mathsf{HonAuth}] \geq \Pr[\mathsf{GoodDec}_{\mathcal{A}} \mid \mathsf{HonAuth}] - \mathsf{negl}(\lambda)$ and $\Pr[\mathsf{BadTr}_{\mathcal{A}}] \leq \mathsf{negl}(\lambda)$. As these hold for every inverse-polynomial error bound ϵ (the parameters z, ζ of Trace depend on ϵ), Π is $(1/p, \mathsf{negl})$-traceable for every polynomial p, as required. □

6.2.4 Putting It All Together

The (γ, α)-risky-limited index-based embedded-identity MA-AB-TT scheme of Construction 6.3 is not yet fully traceable: the limitedness threshold α is hardcoded into Trace (so only decoders with constant advantage $\geq \alpha$ are guaranteed to be traced), and tracing succeeds only with constant probability γ rather than with overwhelming probability. The following two corollaries strip these restrictions in turn—yielding a fully traceable (but still index-based embedded-identity) MA-AB-TT scheme—by chaining the riskiness mitigation compiler of Construction 6.8 and the limitedness mitigation compiler of Construction 6.13.

Corollary 6.17 (Limited index-based embedded-identity MA-AB-TT from MA-EIPLBE). *Let Π be the (γ, α)-risky limited index-based embedded-identity MA-AB-TT scheme given in Construction 6.3. By applying the riskiness mitigation compiler of Construction 6.8 to Π with (output) riskiness parameter 1 and repetition parameter $\tau = (1/\gamma) \cdot \omega(\log \lambda)$, we obtain a scheme $\widehat{\Pi}$ that is α-limited (non-risky) index-based embedded-identity MA-AB-TT.*

Proof. By Proposition 6.7, Π is (γ, α)-risky limited, i.e., γ-risky and α-limited index-based embedded-identity MA-AB-TT. Since $\gamma < 1$, the choice $1 \in (\gamma, 1]$ is admissible in Proposition 6.11, which instantiates the proposition with input riskiness γ and output riskiness 1, and the corresponding repetition (i.e., copy) parameter $\tau = (1/\gamma) \cdot \omega(\log \lambda)$. Hence, the compiled scheme $\widehat{\Pi}$ is an α-limited (non-risky) index-based embedded-identity MA-AB-TT scheme by definition (Definition 6.1). Moreover, the IND-CPA security of $\widehat{\Pi}$ follows from Proposition 6.10, since the IND-CPA security of Π is itself inherited from MA-EIPLBE scheme Π' via Proposition 6.6. □

Corollary 6.18 (Fully traceable index-based embedded-identity MA-AB-TT from MA-EIPLBE). *Let $\widehat{\Pi}$ be the α-limited index-based embedded-identity MA-AB-TT scheme of Corollary 6.17. By applying the limitedness mitigation compiler of Construction 6.13 to $\widehat{\Pi}$ with the hardcoded limitedness constant α, we obtain a scheme $\widetilde{\Pi}$ that*

is (non-risky, non-limited) fully traceable index-based embedded-identity MA-AB-TT. In particular,

$$\underbrace{\Pi'}_{\text{MA-EIPLBE}} \xrightarrow{\text{Construction 6.3}} \underbrace{\Pi}_{(\gamma,\alpha)\text{-risky-limited}} \xrightarrow{\text{Construction 6.8}} \underbrace{\widehat{\Pi}}_{\alpha\text{-limited}} \xrightarrow{\text{Construction 6.13}} \underbrace{\widetilde{\Pi}}_{\text{non-risky, non-limited}},$$

hence, any MA-EIPLBE that is decoder-based message hiding, decoder-based predicate hiding, and weak decoder-based mode indistinguishable yields a fully traceable index-based embedded-identity MA-AB-TT scheme with the same access-structure expressiveness.

Proof. By Corollary 6.17, $\widehat{\Pi}$ is α-limited with the constant α hardcoded into its Trace algorithm. Applying Proposition 6.16 with $\epsilon \coloneqq \alpha$ then yields a (non-limited) traceable index-based embedded-identity MA-AB-TT scheme $\widetilde{\Pi}$. IND-CPA security of $\widetilde{\Pi}$ follows from Proposition 6.15, since IND-CPA security of $\widehat{\Pi}$ was established in Corollary 6.17. Finally, neither compiler alters the way the tracing-side policy $\mathbb{T}$ and the attribute-side policy $\mathbb{A}$ enter Enc, TrKeyGen or AttKeyGen—both compilers operate componentwise across independent parallel copies—so $\widetilde{\Pi}$ inherits the full access-structure expressiveness of Π'. □

Finally, we can extend the results of [GKW19], which showed how to generically turn an *index-based* embedded-identity scheme into a *full* embedded-identity scheme in the single-authority setting, to the multi-authority setting.

Corollary 6.19 (Fully traceable embedded-identity MA-AB-TT). *Assuming the existence of collision-resistant hash function H, if there exists a fully traceable index-based embedded-identity MA-AB-TT scheme $\widetilde{\Pi}$, then there exists a fully traceable (non-index-based) MA-AB-TT scheme $\overline{\Pi}$ with $\mathcal{GID} = \{0,1\}^\lambda$.*

Proof (sketch). We adapt the generic index-stripping compiler of [GKW19], in the single-authority setting, to the multi-authority attribute-based setting. In [GKW19], the index-stripping step proceeds in two stages: first from index-based to a bounded scheme [GKW19, Section 9], then from bounded to unbounded [GKW19, Section 10]. We give a single-step variant that uses a public collision-resistant hash function (CRHF) to derive each user's index deterministically, which collapses both stages because CRHF security directly yields pairwise-distinct indices for any polynomial number of registered users, sidestepping the parallel repetition constructions of [GKW19].

Concretely, let $H\colon \{0,1\}^\lambda \to [2^\kappa]$ be sampled from a CRHF family, with $\kappa = \omega(\log\lambda)$ such that no PPT adversary can find a collision in H except with negligible probability. Given a fully traceable index-based embedded-identity MA-AB-TT scheme $\widetilde{\Pi} = (\widetilde{\mathsf{GlobalSetup}}, \widetilde{\mathsf{TrAuthSetup}}, \widetilde{\mathsf{AttAuthSetup}}, \widetilde{\mathsf{TrKeyGen}}, \widetilde{\mathsf{AttKeyGen}}, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{Dec}}, \widetilde{\mathsf{Trace}})$ with identifier space $\mathcal{GID}_{\kappa,\lambda} = [2^\kappa] \times \{0,1\}^\lambda$, define the compiled scheme $\overline{\Pi}$ with identifier space $\mathcal{GID} = \{0,1\}^\lambda$ by including the description of H in gp and inheriting all algorithms of $\widetilde{\Pi}$ except for the key-generation procedures, which become

$$\overline{\mathsf{TrKeyGen}}(\mathsf{gp}, \mathsf{msk}_{\mathsf{tid}}, \mathsf{gid}) \coloneqq \widetilde{\mathsf{TrKeyGen}}\big(\mathsf{gp}, \mathsf{msk}_{\mathsf{tid}}, (H(\mathsf{gid}), \mathsf{gid})\big),$$
$$\overline{\mathsf{AttKeyGen}}(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, \mathsf{gid}) \coloneqq \widetilde{\mathsf{AttKeyGen}}\big(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, (H(\mathsf{gid}), \mathsf{gid})\big).$$

Determinism of H ensures that all authorities consistently agree on the index $H(\mathsf{gid})$ assigned to each gid without any coordination. Since the global identifiers $\mathsf{gid}_1, \ldots, \mathsf{gid}_N \in \{0,1\}^\lambda$ registered during the experiment are explicit and distinct, any pair $\mathsf{gid}_i \neq \mathsf{gid}_j$ with $H(\mathsf{gid}_i) = H(\mathsf{gid}_j)$ would constitute an explicit CRHF collision. Therefore, the distinct-indices convention of Section 6.2.1 is satisfied except with negligible probability. Moreover, correctness, IND-CPA security and traceability of $\overline{\Pi}$ then follow from those of $\widetilde{\Pi}$ modulo this negligible collision event. □

Putting everything together, we obtain the following theorem.

Theorem 6.20 (Secret-key MA-AB-TT). *Assuming MDDH_k in prime-order pairing groups, there exists a secret-key MA-AB-TT scheme that achieves adaptive IND-CPA security and traceability against static corruption of authorities in the random oracle model. The scheme supports arbitrary monotone access structures for both access control and tracing policies and is defined over authority universes $\mathcal{AID}, \mathcal{TID}$ and user universes $\mathcal{GID}$ of exponential size. Global parameters, user secret keys, and ciphertexts all have size $\mathsf{poly}(\lambda, \log(|\mathcal{AID}|), \log(|\mathcal{TID}|), \log(|\mathcal{GID}|))$.*

6.3 Construction of Public-Key MA-AB-TT

The secret-key construction of Section 6.2 produces the tracing ciphertexts with the secret-mode algorithm EncSK' of MA-EIPLBE, and therefore, the Trace algorithm requires the master secret key $\mathsf{msk}_{\mathsf{tid}^*}$ of the designated tracing authority. As already observed in Section 5, this is the only reason tracing is restricted to the authority itself, and whenever the public-mode encryption EncPK' is expressive enough to encrypt under a nontrivial identity predicate f (i.e., the underlying MA-2ABE is *not* mixed), the secret-mode algorithm becomes redundant and the index ciphertexts that drive tracing can be produced from public keys alone. This immediately yields *public* traceability: any party can run Trace using only the public keys, and the tracing key is $\mathsf{tk}_{\mathsf{tid}^*} = \mathsf{pk}_{\mathsf{tid}^*}$ in the sense of Definition 6.1.

We make this precise by instantiating the tracing-side predicate class with the comparison predicates $\mathcal{F}_N^{>}$ given in Appendix A, rather than the embedded-identity predicates of Section 5.4. The price of this added public-mode expressiveness is that the parameters grow with the number of users N rather than with $\log N$ (this is inherent to comparison predicates over N users already in the single-authority setting [BW06, BCFG17]). The resulting scheme is therefore an *index-based* (bounded-user) MA-AB-TT with public traceability.

Building block. We instantiate the construction with a (plain) MA-2ABE scheme

$$\Pi' = (\mathsf{GlobalSetup}', \mathsf{AuthSetup}', \mathsf{KeyGen}', \mathsf{EncPK}', \mathsf{EncSK}', \mathsf{Dec}'),$$

over the authority space $\mathcal{ID} \coloneqq \mathcal{TID} \cup \mathcal{AID}$ for the user space $\mathcal{GID} \coloneqq [N]$ and the comparison predicate class $\mathcal{F}_N^{>} = \{(> i)\}_{i\in[0;N]}$, where $(> i)(j) = 1$ if and only if $j > i$. Crucially, Π' is *not* mixed, i.e., its public-mode algorithm EncPK' is defined for every predicate $(> i)$, not only for the all-accepting $(> 0) = \mathbf{1}$. This is precisely what lets the tracing probes be produced in the *public* mode. We require Π' to be decoder-based mode indistinguishable, decoder-based predicate hiding, and decoder-based message hiding (Definition 5.10), and oblivious to id^* (Definition 5.4). We stress that this differs from the secret-key setting of Section 6.2, where there the tracing probes are secret-mode (EncSK') ciphertexts, and hence, the decoder-based predicate- and message-hiding notions of Definition 5.10—which are formulated for EncSK'—apply to them directly. Whereas here the probes are *public*-mode (EncPK') ciphertexts, and we transfer those notions to the probes through (decoder-based) mode indistinguishability, which for the non-mixed Π' relates EncSK' and EncPK' on every predicate $(> i)$.

High-level idea. The construction is identical to Construction 6.3, with two changes. First, the user identifier is a plain index $\mathsf{gid} = j \in [N]$ (there is no embedded-identity component), and the tracing algorithm probes the comparison predicates $(> i)$ for $i \in [0; N]$ rather than the embedded-identity predicates $f_{t,i}$. Second, and more importantly, the tracing probes are produced with the *public-mode* algorithm $\mathsf{EncPK}'(\mathsf{gp}, \mathsf{pk}_{\mathsf{tid}^*}, \cdot, \pi = ((> i), \mathbb{A}^\star), \cdot)$, designating the traced authority tid^* as the special MA-2ABE authority id^* through its *public* key $\mathsf{pk}_{\mathsf{tid}^*}$. Since no master secret key is used, tracing is public. As in Construction 6.3, the tracing policy $\mathbb{T}$ and attribute policy $\mathbb{A}$ are folded into the conjunctive product access structure $\mathbb{T} \boxtimes \mathbb{A}$.

Construction 6.21 (Public-key index-based MA-AB-TT). The construction is parameterized by the number of users $N = N(\lambda)$ and uses the MA-2ABE scheme Π' described above. With $\mathbb{T} \boxtimes \mathbb{A}$ defined as in Construction 6.3, the index-based MA-AB-TT scheme Π over $\mathcal{GID} = [N]$ is defined as follows:

- $\mathsf{GlobalSetup}(1^\lambda)$ outputs $\mathsf{gp} \leftarrow \mathsf{GlobalSetup}'(1^\lambda)$.
- $\mathsf{TrAuthSetup}(\mathsf{gp}, \mathsf{tid})$ outputs $(\mathsf{pk}_{\mathsf{tid}}, \mathsf{msk}_{\mathsf{tid}}) \leftarrow \mathsf{AuthSetup}'(\mathsf{gp}, \mathsf{tid})$ for $\mathsf{tid} \in \mathcal{TID}$.
- $\mathsf{AttAuthSetup}(\mathsf{gp}, \mathsf{aid})$ outputs $(\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}}) \leftarrow \mathsf{AuthSetup}'(\mathsf{gp}, \mathsf{aid})$ for $\mathsf{aid} \in \mathcal{AID}$.
- $\mathsf{TrKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{tid}}, j)$ outputs $\mathsf{sk}_{\mathsf{tid},j} \leftarrow \mathsf{KeyGen}'(\mathsf{gp}, \mathsf{msk}_{\mathsf{tid}}, j)$ for $j \in [N]$.
- $\mathsf{AttKeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, j)$ outputs $\mathsf{sk}_{\mathsf{aid},j} \leftarrow \mathsf{KeyGen}'(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, j)$ for $j \in [N]$.
- $\mathsf{Enc}(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu)$ designates $\mathsf{id}^* \coloneqq \min(T \cup A)$ (the lexicographically smallest authority identifier in $T \cup A$) and outputs $\mathsf{ct} \leftarrow \mathsf{EncPK}'(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in(T\cup A)\setminus\{\mathsf{id}^*\}}, \pi = ((> 0), \mathbb{T} \boxtimes \mathbb{A}), \mu)$, where $(> 0) = \mathbf{1}$ accepts every user index.
- $\mathsf{Dec}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{tid},j}\}_{\mathsf{tid}\in T'}, \{\mathsf{sk}_{\mathsf{aid},j}\}_{\mathsf{aid}\in A'}, \mathsf{ct})$ outputs $\mu \leftarrow \mathsf{Dec}'(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{id},j}\}_{\mathsf{id}\in T'\cup A'}, \mathsf{ct})$.

- $\mathsf{Trace}^D(\mathsf{gp}, 1^{\lceil 1/\epsilon \rceil}, \mathsf{pk}_{\mathsf{tid}^*}, \{\mathsf{pk}_{\mathsf{tid}}\}_{\mathsf{tid}\in T}, \mathbb{T}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathbb{A}, \mu_0, \mu_1)$ proceeds as follows. Set $\mathbb{A}^\star \coloneqq \mathbb{T} \boxtimes \mathbb{A}$, $\mathsf{id}^* \coloneqq \mathsf{tid}^*$, and $y \coloneqq \lceil 1/\epsilon \rceil$. For each $i \in [0; N]$, form an empirical estimate

$$\widetilde{p}_i \coloneqq \frac{1}{W}\left|\{k \in [W] \mid D(\mathsf{ct}_{i,k}) = b_{i,k}\}\right|,$$

where each $\mathsf{ct}_{i,k} \leftarrow \mathsf{EncPK}'\big(\mathsf{gp}, \mathsf{pk}_{\mathsf{tid}^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in(T\cup A)\setminus\{\mathsf{tid}^*\}}, \pi = ((>i), \mathbb{A}^\star), \mu_{b_{i,k}}\big)$ is sampled with a fresh $b_{i,k} \xleftarrow{\$} \{0,1\}$, over $W \coloneqq \lceil \lambda N^2 y^2 \rceil$ samples. Set $\widetilde{\mu}_i \coloneqq \widetilde{p}_{i-1} - \widetilde{p}_i$ for $i \in [N]$ and output

$$J_{\mathsf{tid}^*} \coloneqq \left\{j \in [N] \mid \widetilde{\mu}_j \geq \frac{1}{4Ny}\right\}.$$

Crucially, Trace invokes only EncPK' on the public keys $\{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in T\cup A}$ (with tid^* designated through $\mathsf{pk}_{\mathsf{tid}^*}$), and hence, uses no master secret key. Therefore, the scheme is *publicly* traceable in the sense of Definition 6.1.

Theorem 6.22 (Publicly traceable MA-AB-TT). *If Π' is a correct, (non-mixed) MA-2ABE scheme for $\mathcal{F}_N^{>}$ that is decoder-based mode indistinguishable, decoder-based predicate hiding, decoder-based message hiding (Definition 5.10), and oblivious to id^* (Definition 5.4), then Π of Construction 6.21 is a correct, IND-CPA secure, and traceable index-based MA-AB-TT scheme with public traceability.*

Proof (sketch). We sketch the proofs of correctness, IND-CPA security and traceability.

Correctness. This follows immediately from the public-mode correctness of Π' and the fact that $(>0)(j) = 1$ for all $j \in [N]$: a key set $\{\mathsf{sk}_{\mathsf{id},j}\}_{\mathsf{id}\in T'\cup A'}$ with $T' \in \mathbb{T}$ and $A' \in \mathbb{A}$ is authorized under $\mathbb{A}^\star = \mathbb{T} \boxtimes \mathbb{A}$, and hence, Dec' recovers μ from $\mathsf{EncPK}'(\cdot, ((>0), \mathbb{A}^\star), \mu)$.

IND-CPA security. With $\mathsf{id}_0^* \coloneqq \min(T \cup A)$, the challenge ciphertext is computed as

$$\mathsf{EncPK}'\big(\mathsf{gp}, \mathsf{pk}_{\mathsf{id}_0^*}, \{\mathsf{pk}_{\mathsf{id}}\}_{\mathsf{id}\in(T\cup A)\setminus\{\mathsf{id}_0^*\}}, ((>0), \mathbb{A}^\star), \mu_b\big),$$

i.e., a public-mode encryption under the all-accepting predicate $(>0) = \mathbf{1}$. The proof argument follows that of Proposition 6.6. Concretely, obliviousness to id^* (Definition 5.4) re-designates the special authority from id_0^* to an honest $\mathsf{id}^* \in A \setminus (C_{\mathsf{att}} \cup C'_{\mathsf{att}})$ (note that the degenerate case $A \subseteq C_{\mathsf{att}} \cup C'_{\mathsf{att}}$ forces $\mathbb{A}^\star$ to be unsatisfiable and renders Π trivially secure), after which mode indistinguishability and message hiding of Π' yield the hybrid chain $\mathsf{EncPK}'((>0), \mu_0) \approx_c \mathsf{EncSK}'((>0), \mu_0) \approx_c \mathsf{EncSK}'((>0), \mu_1) \approx_c \mathsf{EncPK}'((>0), \mu_1)$. In particular, the message is hidden through the secret mode—exactly as in the secret-key construction—and hence, the mode-indistinguishability hops are needed here as well. The only thing that changes in the public setting is that the *tracing* probes, not the IND-CPA challenge, move to the public mode.

Traceability. Fix a tracing authority $\mathsf{tid}^* \in H_{\mathsf{trace}}$. The probes are produced in the public mode as $\mathsf{EncPK}'(\cdot, \pi = ((>i), \mathbb{A}^\star), \cdot)$. By decoder-based mode indistinguishability of Π', which for the non-mixed Π' holds on every $(>i)$ (Theorem 5.11), replacing each probe by the corresponding secret-mode ciphertext $\mathsf{EncSK}'(\cdot, \pi = ((>i), \mathbb{A}^\star), \cdot)$ perturbs the decoder's per-index advantage by at most a negligible amount. Note that the decoder-based notions of Definition 5.10 are stated for the secret mode EncSK' while Trace measures public-mode (EncPK') quantities, so we elaborate next why this holds in the public mode too. Fix the pirate decoder D output by the tracing adversary. Crucially, D is produced *before* any probe is sampled, and therefore, the reduction can feed the *same* D to the mode-indistinguishability, predicate-hiding, and message-hiding games. More precisely, write p_i^{pk} for the public-mode value actually estimated by $\widetilde{p}_i$ and p_i^{sk} for its secret-mode analog (the same message-distinguishing quantity computed from $\mathsf{EncSK}'(\cdot, ((>i), \mathbb{A}^\star), \cdot)$). For a non-traitor j, the no-false-trace step needs the measured gap $|p_{j-1}^{\mathsf{pk}} - p_j^{\mathsf{pk}}|$ to be negligible, which follows from the triangle inequality

$$\left|p_{j-1}^{\mathsf{pk}} - p_j^{\mathsf{pk}}\right| \leq \underbrace{\left|p_{j-1}^{\mathsf{pk}} - p_{j-1}^{\mathsf{sk}}\right|}_{\text{mode ind. at } (>j-1)} + \underbrace{\left|p_{j-1}^{\mathsf{sk}} - p_j^{\mathsf{sk}}\right|}_{\text{predicate hiding}} + \underbrace{\left|p_j^{\mathsf{sk}} - p_j^{\mathsf{pk}}\right|}_{\text{mode ind. at } (>j)} \leq \mathsf{negl}(\lambda),$$

where the two outer terms are bounded by decoder-based mode indistinguishability applied at the fixed predicates $(>j-1)$ and $(>j)$ (for each message μ_0, μ_1 separately), and the middle term by decoder-based predicate hiding, using the fact that $(>j-1)$ and $(>j)$ differ only on index j, and therefore, agree on every queried user iff j is not a traitor. The correct-trace step is identical, bridging p_N^{pk} to p_N^{sk} at $(>N)$ and invoking decoder-based

message hiding. Two points make this compatible with the decoder-based setting. First, each mode switch is at a *single* fixed predicate ($> i$), where the two modes are functionally identical to every user ($j > i$ decrypts both to the message, $j \leq i$ gets $\perp$ from both). Hence, the *strong* decoder-based mode indistinguishability of Theorem 5.11, holds on every ($> i$) for the non-mixed Π', and requires no admissibility restriction on the queried users—in contrast to the *weak* cross-predicate bridge of the secret-key construction, which is what forces the riskiness/limitedness there. Second, a union bound over the $N + 1$ indices keeps the accumulated gap negligible. After this bridge, the per-tid* analysis is the index-based private linear broadcast tracing argument of [GKW18]. Concretely, writing p_i for the value estimated by $\widetilde{p}_i$ and $\mu_i \coloneqq p_{i-1} - p_i$, a Hoeffding bound over the $W = \lceil \lambda N^2 y^2 \rceil$ samples gives $|\widetilde{\mu}_i - \mu_i| < 1/(8Ny)$ for all i except with probability $2^{-\Omega(\lambda)}$. Then, the two bounds follow:

- **No false trace:** For any j for which the adversary does not hold an authorized key set, decoder-based predicate hiding (Definition 5.10) applied to ($> j-1$) and ($> j$) (which differ only on index j) yields $\mu_j \leq \mathsf{negl}(\lambda)$. Combined with the Hoeffding bound $|\widetilde{\mu}_j - \mu_j| < 1/(8Ny)$ above, this gives $\Pr[\widetilde{\mu}_j \geq 1/(4Ny)] \leq \mathsf{negl}(\lambda)$. Hence, $J_{\mathsf{tid}^*} \subseteq J$ except with negligible probability, establishing $\Pr[\mathsf{BadTr}] \leq \mathsf{negl}(\lambda)$.
- **Correct trace:** Decoder-based message hiding (Definition 5.10) at ($> N$) (which rejects every $j \in [N]$) gives $p_N \leq \mathsf{negl}(\lambda)$, while p_0 equals the decoder's advantage on real ciphertexts. Since the real ciphertext and the $i = 0$ probe are both public-mode encryptions of (> 0) $= \mathbf{1}$—and, by obliviousness to id^* (Definition 5.4) applied at the all-accepting predicate $\mathbf{1}$, re-designating the special authority from $\mathsf{id}^* = \min(T \cup A)$ (used by Enc) to tid^* (used by the probe) leaves this distribution unchanged—we have that $p_0 \geq \epsilon - \mathsf{negl}(\lambda)$ on GoodDec. Summing, $\sum_{i \in [N]} \mu_i = p_0 - p_N \geq \epsilon - \mathsf{negl}(\lambda)$, so some $\mu_{j^\star} \geq (\epsilon - \mathsf{negl}(\lambda))/N \geq 1/(2Ny)$. By the same Hoeffding bound, $\widetilde{\mu}_{j^\star} > 1/(2Ny) - 1/(8Ny) > 1/(4Ny)$, and therefore, the corresponding user $j^\star$ is registered and traced, i.e., $\emptyset \neq J_{\mathsf{tid}^*} \subseteq J$.

The aggregation of the per-tid* sets $\{J_{\mathsf{tid}^*}\}_{\mathsf{tid}^* \in H_{\mathsf{trace}}}$ into the multi-tid events GoodTr and BadTr is identical to that of Proposition 6.7. But unlike the embedded-identity compiler of Construction 6.3, this index-based comparison tracing yields full traceability directly. This is because the real ciphertext and the $i = 0$ probe are in the same public mode and, by obliviousness to id^*, encrypt identically under the all-accepting predicate $\mathbf{1}$, GoodDec coincides with p_0 exactly (no mode-indistinguishability gap is incurred at this step) and the comparison predicates admit strong index hiding. Hence, no riskiness or limitedness issues arise, and no mitigation compilers are needed.

Public traceability. By inspection, Trace uses no secret key and the only key material it touches is the *public* key $\mathsf{pk}_{\mathsf{tid}^*}$ (used to designate $\mathsf{id}^* = \mathsf{tid}^*$ inside EncPK′). Hence, the tracing key is $\mathsf{tk}_{\mathsf{tid}^*} = \mathsf{pk}_{\mathsf{tid}^*}$, and tracing can be executed by any party. □

Instantiation. Plugging the predicate-hiding AB-sIPFE for comparisons of Theorem A.3 into the MA-2ABE construction of Section 5.2 yields a non-mixed MA-2ABE for $\mathcal{F}_N^{>}$ whose public-mode encryption supports every ($> i$), with public keys and ciphertexts of size $O(N)$. By Proposition 5.8 (applied to the comparison AB-sIPFE of Proposition A.2), this scheme is 1-bounded mode indistinguishable and oblivious to id^*, and is 1-bounded predicate- and message-hiding. Note that these three notions lift to their decoder-based counterparts given in Definition 5.10 via Theorems 5.11 to 5.13, respectively. Putting everything together we obtain the following.

Corollary 6.23 (Public-key MA-AB-TT). *Assuming* MDDH_k *in prime-order pairing groups, there exists an index-based MA-AB-TT scheme with* public *traceability and adaptive IND-CPA security against static corruption of authorities in the random oracle model. The scheme supports arbitrary monotone access structures for both access control and tracing policies, is defined over authority universes* $\mathcal{AID}, \mathcal{TID}$ *of exponential size and a bounded user universe* $\mathcal{GID} = [N]$*, and has global parameters and user secret keys of size* $\mathsf{poly}(\lambda)$ *and authority public keys and ciphertexts of size* $\mathsf{poly}(\lambda) \cdot N$.

The linear dependence on N is inherited entirely from the AB-sIPFE for comparison predicates. Hence, any AB-sIPFE for $\mathcal{F}_N^{>}$ with smaller ciphertexts (e.g., the $O(\sqrt{N})$ approach of [BCFG17] discussed in Appendix A) would immediately improve the parameters of Corollary 6.23.

Acknowledgements

The work of Robert Schädlich is supported by EPSRC grant EP/Y02432X/1. The work of Erkan Tairi is supported by the Austrian Science Fund (FWF) Project J4879-N, and research grants from Bakar Funds, Peder Sather Funds and Input Output Global.

References

[ABDP15] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 733–751. Springer, Berlin, Heidelberg, March / April 2015. 6

[ACGU20] Michel Abdalla, Dario Catalano, Romain Gay, and Bogdan Ursu. Inner-product functional encryption with fine-grained access control. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 467–497. Springer, Cham, December 2020. 6

[AG21] Miguel Ambrona and Romain Gay. Multi-authority ABE, revisited. Cryptology ePrint Archive, Report 2021/1381, 2021. 10

[AG23] Miguel Ambrona and Romain Gay. Multi-authority ABE for non-monotonic access structures. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part II*, volume 13941 of *LNCS*, pages 306–335. Springer, Cham, May 2023. 3, 6, 10, 11

[AGT21] Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-party functional encryption. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part II*, volume 13043 of *LNCS*, pages 224–255. Springer, Cham, November 2021. 6

[ALMT20] Shweta Agrawal, Benoît Libert, Monosij Maitra, and Radu Titiu. Adaptive simulation security for inner product functional encryption. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 34–64. Springer, Cham, May 2020. 11

[ALS16] Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 333–362. Springer, Berlin, Heidelberg, August 2016. 6

[ATY23] Shweta Agrawal, Junichi Tomida, and Anshu Yadav. Attribute-based multi-input FE (and more) for attribute-weighted sums. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 464–497. Springer, Cham, August 2023. 6

[AWY20] Shweta Agrawal, Daniel Wichs, and Shota Yamada. Optimal broadcast encryption from LWE and pairings in the standard model. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 149–178. Springer, Cham, November 2020. 10

[BBF+25] Rishiraj Bhattacharyya, Jan Bormet, Sebastian Faust, Pratyay Mukherjee, and Hussien Othman. CCA-secure traceable threshold (ID-based) encryption and application. In Chun-Ying Huang, Jyh-Cheng Chen, Shiuh-Pyng Shieh, David Lie, and Véronique Cortier, editors, *ACM CCS 2025*, pages 2324–2338. ACM Press, October 2025. 3, 7

[BBH06] Dan Boneh, Xavier Boyen, and Shai Halevi. Chosen ciphertext secure public key threshold encryption without random oracles. In David Pointcheval, editor, *CT-RSA 2006*, volume 3860 of *LNCS*, pages 226–243. Springer, Berlin, Heidelberg, February 2006. 3

[BCFG17] Carmen Elisabetta Zaira Baltico, Dario Catalano, Dario Fiore, and Romain Gay. Practical functional encryption for quadratic functions with applications to predicate encryption. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 67–98. Springer, Cham, August 2017. 16, 33, 66, 68

[BDV20] Luís T. A. N. Brandão, Michael Davidson, and Apostol Vassilev. Nist roadmap toward criteria for threshold schemes for cryptographic primitives. NIST Internal Report 8214A, National Institute of Standards and Technology, 2020. 3

[Bei96] Amos Beimel. *Secure Schemes for Secret Sharing and Key Distribution.* PhD thesis, Technion - Israel Institute of Technology, Haifa, Israel, 1996. https://www.cs.bgu.ac.il/~beimel/Papers/thesis.pdf. 17

[BFO26] Jan Bormet, Sebastian Faust, and Hussien Othman. Threshold traitor tracing revisited: Insider attacks and multi-traitor tracing. Cryptology ePrint Archive, Report 2026/422, 2026. 7

[BHO25] Jan Bormet, Jonas Hofmann, and Hussien Othman. Traceable threshold encryption without a trusted dealer. In Goichiro Hanaoka and Bo-Yin Yang, editors, *ASIACRYPT 2025, Part VI*, volume 16250 of *LNCS*, pages 471–505. Springer, Singapore, December 2025. 3, 7

[BL90] Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In Shafi Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 27–35. Springer, New York, August 1990. 17, 18

[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, *ASIACRYPT 2024, Part III*, volume 15486 of *LNCS*, pages 33–66. Springer, Singapore, December 2024. 7

[BMV19] Luís T. A. N. Brandão, Nicky Mouha, and Apostol Vassilev. Threshold schemes for cryptographic primitives: Challenges and opportunities in standardization and validation of threshold cryptography. NIST Internal Report 8214, National Institute of Standards and Technology, 2019. 3

[BN08] Dan Boneh and Moni Naor. Traitor tracing with constant size ciphertext. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM CCS 2008*, pages 501–510. ACM Press, October 2008. 7

[Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 31–46. Springer, Berlin, Heidelberg, January 2003. 3

[BP08] Olivier Billet and Duong Hieu Phan. Efficient traitor tracing from collusion secure codes. In Reihaneh Safavi-Naini, editor, *ICITS 08*, volume 5155 of *LNCS*, pages 171–182. Springer, Berlin, Heidelberg, August 2008. 7

[BP26] Luís T. A. N. Brandão and René Peralta. Nist first call for multi-party threshold schemes. NIST Internal Report 8214C, National Institute of Standards and Technology, 2026. 3

[BPR24] Dan Boneh, Aditi Partap, and Lior Rotem. Accountability for misbehavior in threshold decryption via threshold traitor tracing. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part VII*, volume 14926 of *LNCS*, pages 317–351. Springer, Cham, August 2024. 3, 7

[BSW06] Dan Boneh, Amit Sahai, and Brent Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 573–592. Springer, Berlin, Heidelberg, May / June 2006. 3, 7, 9

[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*, pages 211–220. ACM Press, October / November 2006. 7, 16, 33, 66

[BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 480–499. Springer, Berlin, Heidelberg, August 2014. 3, 7

[CC09] Melissa Chase and Sherman S. M. Chow. Improving privacy and security in multi-authority attribute-based encryption. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *ACM CCS 2009*, pages 121–130. ACM Press, November 2009. 3

[CFN94] Benny Chor, Amos Fiat, and Moni Naor. Tracing traitors. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 257–270. Springer, Berlin, Heidelberg, August 1994. 3, 7

[CG99] Ran Canetti and Shafi Goldwasser. An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 90–106. Springer, Berlin, Heidelberg, May 1999. 3

[CGJ+99] Ran Canetti, Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Adaptive security for threshold cryptosystems. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 98–115. Springer, Berlin, Heidelberg, August 1999. 3

[Cha07] Melissa Chase. Multi-authority attribute based encryption. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 515–534. Springer, Berlin, Heidelberg, February 2007. 3

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer, Berlin, Heidelberg, May 2001. 3

[CMMS25] Anirban Chakrabarti, Monosij Maitra, Arup Mondal, and Kushaz Sehgal. Silent threshold traitor tracing & enhancing mempool privacy. In Chun-Ying Huang, Jyh-Cheng Chen, Shiuh-Pyng Shieh, David Lie, and Véronique Cortier, editors, *ACM CCS 2025*, pages 1769–1783. ACM Press, October 2025. 7

[CPP05] Hervé Chabanne, Duong Hieu Phan, and David Pointcheval. Public traceability in traitor tracing schemes. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 542–558. Springer, Berlin, Heidelberg, May 2005. 7

[CPP25] Sébastien Canard, Nathan Papon, and Duong Hieu Phan. Public traceability in threshold decryption. *CiC*, 2(2):21, 2025. 7

[CVW+18] Yilei Chen, Vinod Vaikuntanathan, Brent Waters, Hoeteck Wee, and Daniel Wichs. Traitor-tracing from LWE made simple and attribute-based. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pages 341–369. Springer, Cham, November 2018. 3, 5, 6, 7, 11

[DDP+25] Sourav Das, Pratish Datta, Aditi Partap, Swagata Sasmal, and Mark Zhandry. Optimal threshold traitor tracing. Cryptology ePrint Archive, Report 2025/2154, 2025. 3, 7, 10, 14, 15, 56, 60

[DF90] Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 307–315. Springer, New York, August 1990. 3

[DKW23] Pratish Datta, Ilan Komargodski, and Brent Waters. Fully adaptive decentralized multi-authority ABE. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 447–478. Springer, Cham, April 2023. 3

[DP23] Pratish Datta and Tapas Pal. Decentralized multi-authority attribute-based inner-product FE: Large universe and unbounded. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 587–621. Springer, Cham, May 2023. 6

[DTV25] Pratish Datta, Junichi Tomida, and Nikhil Vanjani. Fully adaptive decentralized MA-ABE: Simplified, optimized, ASP supported. In Goichiro Hanaoka and Bo-Yin Yang, editors, *ASIACRYPT 2025, Part VI*, volume 16250 of *LNCS*, pages 68–105. Springer, Singapore, December 2025. 3

[EHK+13] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. An algebraic framework for Diffie-Hellman assumptions. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Berlin, Heidelberg, August 2013. 17, 31

[FP01] Pierre-Alain Fouque and David Pointcheval. Threshold cryptosystems secure against chosen-ciphertext attacks. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 351–368. Springer, Berlin, Heidelberg, December 2001. 3

[Gay16] Romain Gay. Functional encryption for quadratic functions, and applications to predicate encryption. Cryptology ePrint Archive, Report 2016/1106, 2016. 16

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013. 7

[GJKR96a] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Robust and efficient sharing of RSA functions. In Neal Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 157–172. Springer, Berlin, Heidelberg, August 1996. 3

[GJKR96b] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Robust threshold DSS signatures. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 354–371. Springer, Berlin, Heidelberg, May 1996. 3

[GKW18] Rishab Goyal, Venkata Koppula, and Brent Waters. Collusion resistant traitor tracing from learning with errors. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *50th ACM STOC*, pages 660–670. ACM Press, June 2018. 3, 6, 7, 10, 11, 12, 19, 21, 32, 33, 45, 46, 68

[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, Part II*, volume 11892 of *LNCS*, pages 149–179. Springer, Cham, December 2019. 3, 7, 8, 16, 48, 49, 52, 65

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *21st ACM STOC*, pages 25–32. ACM Press, May 1989. 15, 60

[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, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 637–668. Springer, Cham, April 2023. 3, 7

[GSY26] Rishab Goyal, Alex Snyder, and Saikumar Yadugiri. Decentralizing traitor tracing: A multi-authority approach. Cryptology ePrint Archive, Report 2026/1076, 2026. 6, 7, 8

[GVW19] Rishab Goyal, Satyanarayana Vusirikala, and Brent Waters. Collusion resistant broadcast and trace from positional witness encryption. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 3–33. Springer, Cham, April 2019. 7

[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, Part III*, volume 15366 of *LNCS*, pages 3–30. Springer, Cham, December 2024. 3

[GY25] Rishab Goyal and Saikumar Yadugiri. Multi-authority encryption with malicious authorities. Cryptology ePrint Archive, Report 2025/412, 2025. 3

[HLL24] Yao-Ching Hsieh, Huijia Lin, and Ji Luo. A general framework for lattice-based ABE using evasive inner-product functional encryption. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part II*, volume 14652 of *LNCS*, pages 433–464. Springer, Cham, May 2024. 10

[KG20] Chelsea Komlo and Ian Goldberg. FROST: Flexible round-optimized Schnorr threshold signatures. In Orr Dunkelman, Michael J. Jacobson, Jr., and Colin O'Flynn, editors, *SAC 2020*, volume 12804 of *LNCS*, pages 34–65. Springer, Cham, October 2020. 3

[LCLS08] Huang Lin, Zhenfu Cao, Xiaohui Liang, and Jun Shao. Secure threshold multi authority attribute based encryption without a central authority. In Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, *INDOCRYPT 2008*, volume 5365 of *LNCS*, pages 426–436. Springer, Berlin, Heidelberg, December 2008. 3

[LCW13] Zhen Liu, Zhenfu Cao, and Duncan S. Wong. Blackbox traceable CP-ABE: How to catch people leaking their keys by selling decryption devices on eBay. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 475–486. ACM Press, November 2013. 5, 6

[Lin17] Huijia Lin. Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 599–629. Springer, Cham, August 2017. 6, 11

[LL20a] Huijia Lin and Ji Luo. Compact adaptively secure ABE from k-Lin: Beyond NC^1 and towards NL. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 247–277. Springer, Cham, May 2020. 6, 8, 11, 12, 13, 14, 18, 21, 24, 26, 28, 29, 76

[LL20b] Huijia Lin and Ji Luo. Succinct and adaptively secure ABE for ABP from k-Lin. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 437–466. Springer, Cham, December 2020. 8

[LLL22] Hanjun Li, Huijia Lin, and Ji Luo. ABE for circuits with constant-size secret keys and adaptive security. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 680–710. Springer, Cham, November 2022. 8, 10, 11

[LLW21] Qiqi Lai, Feng-Hao Liu, and Zhedong Wang. New lattice two-stage sampling technique and its applications to functional encryption - stronger security and smaller ciphertexts. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 498–527. Springer, Cham, October 2021. 6

[LRSW99] Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, and Stefan Wolf. Pseudonym systems. In Howard M. Heys and Carlisle M. Adams, editors, *SAC 1999*, volume 1758 of *LNCS*, pages 184–199. Springer, Berlin, Heidelberg, August 1999. 3

[Luo24] Ji Luo. Ad hoc broadcast, trace, and revoke: Plus time-space trade-offs for attribute-based encryption. *CiC*, 1(2):15, 2024. 7

[LV16] Huijia Lin and Vinod Vaikuntanathan. Indistinguishability obfuscation from DDH-like assumptions on constant-degree graded encodings. In Irit Dinur, editor, *57th FOCS*, pages 11–20. IEEE Computer Society Press, October 2016. 6, 11

[LW11] Allison B. Lewko and Brent Waters. Decentralizing attribute-based encryption. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 568–588. Springer, Berlin, Heidelberg, May 2011. 3, 5, 11, 17, 18, 34, 49

[LW15] Zhen Liu and Duncan S. Wong. Traceable CP-ABE on prime order groups: Fully secure and fully collusion-resistant blackbox traceable. In Sihan Qing, Eiji Okamoto, Kwangjo Kim, and Dongmei Liu, editors, *ICICS 15*, volume 9543 of *LNCS*, pages 109–124. Springer, Cham, December 2015. 5, 6

[MKE09] Sascha Müller, Stefan Katzenbeisser, and Claudia Eckert. Distributed attribute-based encryption. In Pil Joong Lee and Jung Hee Cheon, editors, *ICISC 08*, volume 5461 of *LNCS*, pages 20–36. Springer, Berlin, Heidelberg, December 2009. 3

[NCD+16] Jianting Ning, Zhenfu Cao, Xiaolei Dong, Junqing Gong, and Jie Chen. Traceable CP-ABE with short ciphertexts: How to catch people selling decryption devices on eBay efficiently. In Ioannis G. Askoxylakis, Sotiris Ioannidis, Sokratis K. Katsikas, and Catherine A. Meadows, editors, *ESORICS 2016, Part II*, volume 9879 of *LNCS*, pages 551–569. Springer, Cham, September 2016. 5, 6

[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, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 388–419. Springer, Berlin, Heidelberg, May 2016. 49

[PD21] Tapas Pal and Ratna Dutta. Attribute-based access control for inner product functional encryption from lwe. In *Progress in Cryptology - LATINCRYPT 2021: 7th International Conference on Cryptology and Information Security in Latin America, Bogotá, Colombia, October 6-8, 2021, Proceedings*, pages 127–148, Berlin, Heidelberg, 2021. Springer-Verlag. 6

[PST06] Duong Phan, Reihaneh Safavi-Naini, and Dongvu Tonien. Generic construction of hybrid public key traitor tracing with full-public-traceability. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP 2006, Part II*, volume 4052 of *LNCS*, pages 264–275. Springer, Berlin, Heidelberg, July 2006. 7

[RW15] Yannis Rouselakis and Brent Waters. Efficient statically-secure large-universe multi-authority attribute-based encryption. In Rainer Böhme and Tatsuaki Okamoto, editors, *FC 2015*, volume 8975 of *LNCS*, pages 315–332. Springer, Berlin, Heidelberg, January 2015. 3, 11, 34, 49

[SG98] Victor Shoup and Rosario Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. In Kaisa Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 1–16. Springer, Berlin, Heidelberg, May / June 1998. 3

[Tom23] Junichi Tomida. Unbounded quadratic functional encryption and more from pairings. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 543–572. Springer, Cham, April 2023. 6

[Wee17] Hoeteck Wee. Attribute-hiding predicate encryption in bilinear groups, revisited. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 206–233. Springer, Cham, November 2017. 6

[Wee20] Hoeteck Wee. Functional encryption for quadratic functions from k-lin, revisited. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 210–228. Springer, Cham, November 2020. 26

[WWW22] Brent Waters, Hoeteck Wee, and David J. Wu. Multi-authority ABE from lattices without random oracles. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 651–679. Springer, Cham, November 2022. 3, 5

[Zha20] Mark Zhandry. New techniques for traitor tracing: Size $N^{1/3}$ and more from pairings. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 652–682. Springer, Cham, August 2020. 3, 7, 14, 15, 56, 59, 60, 63, 64

[Zha25] Mark Zhandry. Optimal traitor tracing from pairings. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part III*, volume 15603 of *LNCS*, pages 305–335. Springer, Cham, May 2025. 3, 6, 7, 10, 11, 14, 19, 21, 32, 33, 43, 46, 47, 48, 49, 52, 55

A Construction of Predicate-Hiding AB-sIPFE for Comparisons with O(N)-Size Ciphertexts

For completeness, we provide a predicate-hiding AB-sIPFE for comparison predicates which can be used to instantiate our MA-AB-TT construction in the setting with public traceability. Let $\mathcal{F}_N^{>} \coloneqq \{(> i)\}_{i\in[0;N]}$, where $(> i)\colon [N] \to \{0,1\}$ is defined as

$$(> i)(j) \coloneqq \begin{cases} 0 & \text{if } j \in [i] \\ 1 & \text{if } j \in [i+1;N]. \end{cases}$$

We denote by $\mathbf{e}_i$ the i unit vector in $\mathbb{Z}_p^N$ and, more generally, $\mathbf{e}_{\mathfrak{s}} \coloneqq \sum_{i\in\mathfrak{s}} \mathbf{e}_i$ for $\mathfrak{s} \subseteq N$.

Construction A.1 (Predicate-hiding AB-sIPFE for $\mathcal{F}_N^{>}$). The construction uses an sIPFE scheme $\Pi' = (\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{EncPK}', \mathsf{EncSK}', \mathsf{Dec}')$ based on a pairing group $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, e, p)$. The construction of the AB-sIPFE scheme Π for $\mathcal{F}_N^{>}$ works as follows:

- $\mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathrm{pub}}, \mathfrak{s}_{\mathrm{priv}})$ outputs

$$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}'\big(1^\lambda, \mathfrak{s}'_{\mathrm{pub}} \coloneqq \mathfrak{s}_{\mathrm{pub}} \cup [kN], \mathfrak{s}'_{\mathrm{priv}} \coloneqq \mathfrak{s}_{\mathrm{priv}} \cup \{\mathsf{sim}\}\big).$$

- $\mathsf{KeyGen}(\mathsf{msk}, j, [\![\mathbf{v}]\!]_2)$ samples $\mathbf{r} \leftarrow\!\!\$\ \mathbb{Z}_p^k$ and outputs

$$\mathsf{sk}' \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}, [\![\mathbf{v}']\!]_2 \coloneqq [\![(\mathbf{v}|_{\mathfrak{s}_{\mathrm{pub}}}, \mathbf{e}_j \otimes \mathbf{r}, \mathbf{v}|_{\mathfrak{s}_{\mathrm{priv}}}, 0)]\!]_2\big).$$

- $\mathsf{EncPK}(\mathsf{mpk}, f = (> i), [\![\mathbf{u}_{\mathrm{pub}}]\!]_1)$ samples $\mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_p^k$ and outputs

$$\mathsf{ct} \leftarrow \mathsf{EncPK}'\big(\mathsf{mpk}', [\![\mathbf{u}'_{\mathrm{pub}}]\!]_1 \coloneqq [\![(\mathbf{u}_{\mathrm{pub}}, \mathbf{e}_{[i]} \otimes \mathbf{s})]\!]_1\big).$$

- $\mathsf{EncSK}(\mathsf{msk}, f = (> i), [\![\mathbf{u}]\!]_1)$ samples $\mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_p^k$ and outputs

$$\mathsf{ct} \leftarrow \mathsf{EncSK}'\big(\mathsf{msk}, [\![\mathbf{u}']\!]_1 \coloneqq [\![(\mathbf{u}|_{\mathfrak{s}_{\mathrm{pub}}}, \mathbf{e}_{[i]} \otimes \mathbf{s}, \mathbf{u}|_{\mathfrak{s}_{\mathrm{priv}}}, 0)]\!]_1\big).$$

- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$ outputs

$$[\![d]\!]_t \coloneqq \mathsf{Dec}'(\mathsf{sk}, \mathsf{ct}).$$

Proposition A.2 (Correctness, security and efficiency). *The AB-sIPFE scheme Π in Construction A.1 satisfies the following properties:*

- *If Π' is correct, then so is Π.*
- *If Π' is mode indistinguishable, then Π is q-bounded mode indistinguishable for every polynomial q.*
- *If Π' is IND-CPA secure, then Π is q-bounded predicate hiding for every polynomial q.*
- *If Π' is IND-CPA secure, then Π is q-bounded message hiding for every polynomial q.*

Proof. Correctness of Π follows from the correctness of Π' and the fact that $(\mathbf{e}_j \otimes \mathbf{r}) \cdot (\mathbf{e}_{[i]} \otimes \mathbf{s})^{\mathrm{T}} = 0$ if $j > i$. Mode indistinguishability is immediate from the mode indistinguishability of Π'.

For predicate hiding, assume w.l.o.g. that the adversary picks $f_0 = (> i_0)$ and $f_1 = (> i_1)$ such that $i_0 < i_1$. First, note that the admissibility condition does not permit the adversary to query $O\mathsf{KeyGen}(j, \cdot)$ such that $j \in [i_0; i_1 - 1]$. For all other key generation queries $O\mathsf{KeyGen}(j, [\![\mathbf{v}]\!]_2)$ such that $j \in [N]$, we have $(\mathbf{e}_j \otimes \mathbf{r}) \cdot (\mathbf{e}_{[i_0]} \otimes \mathbf{s})^{\mathrm{T}} = (\mathbf{e}_j \otimes \mathbf{r}) \cdot (\mathbf{e}_{[i_1]} \otimes \mathbf{s})^{\mathrm{T}}$, in which case predicate hiding follows from the IND-CPA security of Π'.

For message hiding, we consider the following sequence of hybrids.

- $\mathsf{Hyb}_0^{(b)}$: This is $\mathsf{ExpMsg}_{\Pi,\mathcal{A}}^{q,b}$ for some polynomial q and challenge bit b. In particular, the challenge ciphertext is computed as

$$\mathsf{ct} \leftarrow \mathsf{EncSK}'\big(\mathsf{msk}, [\![\mathbf{u}'_b]\!]_1 = [\![(\mathbf{u}_b|_{\mathfrak{s}_{\mathrm{pub}}}, \mathbf{e}_{[i^*]} \otimes \mathbf{s}, \mathbf{u}_b|_{\mathfrak{s}_{\mathrm{priv}}}, 0)]\!]_1\big)$$

and upon receiving the κ-th key generation query $O\mathsf{KeyGen}^*(j_\kappa, [\![\mathbf{v}_{\kappa,0}]\!]_2, [\![\mathbf{v}_{\kappa,1}]\!]_2)$ for $\kappa \in [Q]$, the challenger samples $\mathbf{r}_\kappa \leftarrow\!\!\$\ \mathbb{Z}_p^k$ and sends

$$\mathsf{sk} \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}, [\![\mathbf{v}'_{\kappa,b}]\!]_2 = [\![(\mathbf{v}_{\kappa,b}|_{\mathfrak{s}_{\mathrm{pub}}}, \mathbf{e}_{j_\kappa} \otimes \mathbf{r}_\kappa, \mathbf{v}_{\kappa,b}|_{\mathfrak{s}_{\mathrm{priv}}}, 0)]\!]_2\big).$$

- $\mathsf{Hyb}_1^{(b)}$: This is the same as $\mathsf{Hyb}_0^{(b)}$ except that the index i^* is programmed into the secret keys. Specifically, the challenge ciphertext is computed as

$$\mathsf{ct} \leftarrow \mathsf{EncSK}'\big(\mathsf{msk}, [\![\mathbf{u}'_b]\!]_1 = [\![(\mathbf{u}_b|_{\mathfrak{s}_{\mathrm{pub}}}, \mathbf{0}_{kN}, \mathbf{u}_b|_{\mathfrak{s}_{\mathrm{priv}}}, 1)]\!]_1\big)$$

and upon receiving the κ-th key generation query $O\mathsf{KeyGen}^*(j_\kappa, [\![\mathbf{v}_{\kappa,0}]\!]_2, [\![\mathbf{v}_{\kappa,1}]\!]_2)$, the challenger sends

$$\mathsf{sk} \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}, [\![\mathbf{v}'_{\kappa,b}]\!]_2 = [\![(\mathbf{v}_{\kappa,b}|_{\mathfrak{s}_{\mathrm{pub}}}, \mathbf{e}_{j_\kappa} \otimes \mathbf{r}_\kappa, \mathbf{v}_{\kappa,b}|_{\mathfrak{s}_{\mathrm{priv}}}, \underbrace{(\mathbf{e}_{j_\kappa} \otimes \mathbf{r}_\kappa)(\mathbf{e}_{[i^*]} \otimes \mathbf{s})^{\mathrm{T}}}_{=\theta_\kappa})]\!]_2\big).$$

Since inner products between $\mathbf{u}'$ and $\mathbf{v}'$ do not change from $\mathsf{Hyb}_0^{(b)}$ to $\mathsf{Hyb}_1^{(b)}$, it follows that $\mathsf{Hyb}_0^{(b)} \approx_c \mathsf{Hyb}_1^{(b)}$ by the IND-CPA security of Π'.

Note that in $\mathsf{Hyb}_1^{(b)}$, we have

$$\theta_\kappa = \begin{cases} 0 & \text{if } j_\kappa \in [i^*+1; N] \\ \mathbf{r}_\kappa \mathbf{s}^{\mathrm{T}} & \text{if } j_\kappa \in [i^*]. \end{cases}$$

- $\mathsf{Hyb}_2^{(b)}$: This is the same as $\mathsf{Hyb}_1^{(b)}$ except that the challenger picks $\theta_\kappa \leftarrow\!\!\$\ \mathbb{Z}_p$ at random for all $\kappa \in [Q]$ such that $j_\kappa \in [i^*]$.

We have $\mathsf{Hyb}_1^{(b)} \approx_c \mathsf{Hyb}_2^{(b)}$ by the MDDH_k assumption in $\mathbb{G}_2$. Given a challenge $([\![\mathbf{R}]\!]_2, [\![\boldsymbol{\theta}]\!]_2) \in \mathbb{G}_2^{Q\times k} \times \mathbb{G}_2^{Q\times 1}$, parse

$$\mathbf{R} \coloneqq \begin{pmatrix} — & \mathbf{r}_1 & — \\ & \vdots & \\ — & \mathbf{r}_Q & — \end{pmatrix} \quad \text{and} \quad \boldsymbol{\theta} \coloneqq \begin{pmatrix} — & \theta_1 & — \\ & \vdots & \\ — & \theta_Q & — \end{pmatrix}.$$

Then we can observe that the challenger simulates $\mathsf{Hyb}_1^{(b)}$ if $\boldsymbol{\theta} = \mathbf{R}\mathbf{s}$ for $\mathbf{s} \leftarrow\!\!\$\ \mathbb{Z}_p^k$, and it simulates $\mathsf{Hyb}_2^{(b)}$ if $\boldsymbol{\theta} \leftarrow\!\!\$\ \mathbb{Z}_p^{Q\times 1}$.

Moreover, we have that $\mathsf{Hyb}_2^{(0)} \approx_c \mathsf{Hyb}_2^{(1)}$ by the IND-CPA security of Π'. To see this, we need to argue that the condition $\mathbf{v}'_{\kappa,0} \cdot (\mathbf{u}'_0)^{\mathrm{T}} = \mathbf{v}'_{\kappa,1} \cdot (\mathbf{u}'_1)^{\mathrm{T}}$ is satisfied for all $\kappa \in [Q]$. If $j_\kappa \in [i^*+1, N]$, then the admissibility condition asserts that $\mathbf{v}_{\kappa,0}\mathbf{u}_0^{\mathrm{T}} = \mathbf{v}_{\kappa,1}\mathbf{u}_1^{\mathrm{T}}$ which immediately implies that also $\mathbf{v}'_{\kappa,0} \cdot (\mathbf{u}'_0)^{\mathrm{T}} = \mathbf{v}'_{\kappa,1} \cdot (\mathbf{u}'_1)^{\mathrm{T}}$. Conversely, if $j_\kappa \in [i^*]$, then the condition holds after applying the statistical shift $\theta_\kappa \mapsto \theta_\kappa + \mathbf{v}_{\kappa,0}\mathbf{v}_0^{\mathrm{T}} - \mathbf{v}_{\kappa,b}\mathbf{v}_b^{\mathrm{T}}$.

□

If Π' is instantiated with the sIPFE scheme of [LL20a], then we obtain the following theorem.

Theorem A.3. *Assuming MDDH_k on pairings, there exists a predicate-hiding AB-sIPFE supporting $\mathcal{F}_N^{>}$ with the following asymptotic parameters:*

$$|\mathsf{mpk}| = |\mathsf{msk}| = |\mathsf{ct}| = O\big(|\mathfrak{s}_{\mathrm{pub}}| + |\mathfrak{s}_{\mathrm{priv}}| + N\big), \qquad |\mathsf{sk}| = O\big(|\mathfrak{s}_{\mathrm{pub}}| + |\mathfrak{s}_{\mathrm{priv}}|\big),$$

where $O(\cdot)$ hides factors polynomial in λ.

To see that $|\mathsf{sk}|$ is independent of N, we observe that in the sIFPE of [LL20a], a secret key for a vector $[\![\mathbf{v}']\!]_2 = [\![(\mathbf{v}'_{\mathrm{pub}}, \mathbf{v}'_{\mathrm{priv}})]\!]_2$ contains a description of $[\![\mathbf{v}'_{\mathrm{pub}}]\!]_2$ but is otherwise independent of the length of $\mathbf{v}'_{\mathrm{pub}}$. In our case, the vector $[\![\mathbf{v}'_{\mathrm{pub}}]\!]_2$ is fully determined by the tuple $([\![\mathbf{r}]\!]_2, j)$ whose size is independent of N.