Threshold Traitor Tracing with Partial-Insider Resilience*

Jan Bormet Hussien Othman

Technische Universität Darmstadt, Darmstadt, Germany
jan.bormet@tu-darmstadt.de, hussien.othman@gmail.com

Abstract

Threshold traitor tracing (Boneh et al. Crypto'24) addresses collusion in threshold encryption by tracing parties who collude to build illegal decryption devices called decoders. However, the original definition does not capture settings where adversaries can access partial decryptions published during normal system operation. In such settings, decoders sold to external buyers could depend on inputting additional partial decryptions from honest parties. Moreover, colluders may exploit observed partial decryptions when constructing a decoder to evade tracing, or even frame honest parties. Recently, Bormet et al. (TCC'26) addressed this by introducing *threshold traitor tracing in the presence of partial insiders*. Their construction, however, works only in a weaker model, where decoders are assumed to output the full decryption, and colluders are assumed to observe only partial decryptions of *valid ciphertexts*. Additionally, they use a trusted dealer for key generation.

In this work, we present the first construction resilient against partial insiders in the stronger model of distinguishing decoders. Furthermore, our construction does not rely on a trusted dealer and remains traceable when colluders have access to a partial decryption oracle.

As part of this, we show how to generically lift traceability under a partial decryption oracle for *valid ciphertexts* to traceability under a partial decryption oracle for arbitrary ciphertexts using NIZK-PoKs. This transform is of independent interest, directly strengthening prior results that consider only valid-ciphertext partial decryption oracles, and allowing future analyses to focus on the simpler valid-ciphertext setting.

1 Introduction

A recent line of work [11, 20, 24, 5, 23, 31], initiated by Boneh, Partap, and Rotem [11], studies *threshold traitor tracing* (TTT), a primitive for mitigating collusion attacks in threshold encryption [32, 33]. In threshold encryption, a ciphertext can be decrypted by any subset of at least $t \leq n$ parties, while remaining secure against coalitions of at most $t-1$ parties. Threshold encryption has recently received significant attention in the context of encrypted blockchain mempools [3, 35, 30, 19, 16, 1, 10].

A central challenge is collusion: committee members may collude to help an adversary decrypt ciphertexts. Threshold traitor tracing models such attacks via a *stateless decoder* created by an arbitrary coalition of parties and sold to a buyer, who can then decrypt arbitrary ciphertexts. The key guarantee is traceability: given black-box access to the decoder, a tracing algorithm can identify at least one colluder. As observed in [18], this approach relies on the assumption that the buyer is incentivized to reveal the decoder to the tracing authority, i.e., to blow the whistle. The underlying premise, therefore, is that the threat of being caught and punished sufficiently disincentivizes collusion.

Bormet, Faust, and Othman [18] recently identified a limitation in the original model of [11]. That is, tracing guarantees apply only if the decoder is sold to an *external* buyer without access to insider information, such as secret key shares or partial decryptions. However, colluders could instead sell the decoder to a committee member and configure it so that decryption succeeds only when the buyer supplies insider information as auxiliary input for each query. Such a decoder still enables decryption of arbitrary ciphertexts, but the colluders can design it such that tracing may now always implicate the insider rather than the original colluders. Consequently, the insider has no incentive to reveal the decoder to the tracer. To address this, the work of Bormet et al. [18] introduces a strengthened definition called *traceability against insiders*. They model traceability against both *full insiders*, who supply their secret key share as insider information, and *partial insiders*, who supply only a partial decryption of the queried ciphertext.

*This work was partially supported by the European Research Council (ERC) under the European Union's Horizon 2020 and Horizon Europe research and innovation programs (grant CRYPTOLAYER-101044770) and the German Research Foundation (DFG) via the DFG CRC 1119 CROSSING (project S7).

In real-world settings, such as encrypted mempools, an inherent threat is that adversaries may observe the system during normal operation and thus learn partial decryptions of honest parties for certain ciphertexts. From a traceability perspective, this raises two important concerns:

(1) Partial decryption oracles can cause frameability. Observing partial decryption from honest parties might allow colluders to learn sensitive information, enabling them to evade tracing or even frame honest committee members.

(2) Decoders requiring partial decryptions may still be useful to external buyers. In certain scenarios, some honest parties may issue partial decryptions even for ciphertexts intended to remain private. This setting is highly relevant in practice, and recently motivated work on *high-threshold security* [8] for threshold encryption, where the adversary can obtain a limited number of partial decryptions of honest parties on the challenge ciphertext. This is of particular interest in *context-dependent threshold decryption* [8], where ciphertexts should only be revealed when a sufficient number of parties issue partial decryptions *under the same context*, and thus a small number of partial decryptions might be available for various contexts. Consequently, decoders that require partial decryptions as insider information can even be sold to *external buyers*, who can naturally observe such decryptions during system operation and thus use the decoder. As a result, the partial insider model is interesting even without directly selling the decoder to an insider, since colluders may evade tracing even when the decoder is sold to an *external* buyer.

The state-of-the-art. To the best of our knowledge, [18] provides the only construction resilient against partial insiders.[1] Unfortunately, it has three shortcomings. First, it assumes a significantly weaker decoder model in which the decoder outputs the full decryption. In modern traitor tracing, however, the decoder is typically modeled as a distinguisher that learns non-trivial information about the plaintext without necessarily outputting the entire plaintext. This is important because it better captures realistic attacks, where partial information may be sufficient to compromise security. Second, it requires a trusted dealer to distribute secret shares, which is undesirable in threshold settings due to a single point of failure. We note that such a trusted dealer, as discussed in [20], cannot be replaced by standard distributed key generation protocols, since these generally allow coalitions of t or more parties to learn others' shares and thus frame them. Third, it only considers a partial decryption oracle for *valid ciphertexts*, requiring the adversary to specify the encryption randomness, rather than allowing colluders to learn partial decryptions on arbitrary, possibly malformed ciphertexts.

In this paper, we present a construction that resolves all issues mentioned above by building on the Threshold Private Linear Broadcast Encryption (TPLBE) technique, which we briefly introduce next.

Tracing Approach: TPLBE. In this work, we follow the PLBE [14] technique for achieving traceability, which was recently extended to the threshold setting by [20, 31]. In TPLBE, there are two encryption methods: standard encryption $\mathsf{Enc}(\mathsf{pk})$ and trace-encryption $\mathsf{TrEnc}(\mathsf{tk}, i)$. Given the tracing key tk and an index $i \in \{0, \ldots, n\}$, $\mathsf{TrEnc}(\mathsf{tk}, i)$ encrypts a message such that only parties with indices greater than i can decrypt, i.e., parties $\leq i$ are "revoked". To trace a decoder, the tracer queries it on $\mathsf{TrEnc}(0), \mathsf{TrEnc}(1), \ldots, \mathsf{TrEnc}(n)$ and identifies a party $i \in [n]$ as a traitor if the decryption probability drops significantly between $\mathsf{TrEnc}(i-1)$ and $\mathsf{TrEnc}(i)$. Tracing relies on three properties. *Indistinguishability* requires that Enc and $\mathsf{TrEnc}(0)$ are computationally indistinguishable. *Index hiding* requires that any adversary not holding sk_i cannot distinguish $\mathsf{TrEnc}(i-1)$ from $\mathsf{TrEnc}(i)$. *Message privacy* requires that $\mathsf{TrEnc}(n)$ is infeasible to decrypt. Together, these properties imply traceability: the decoder's success probability is initially high (by indistinguishability) and eventually negligible (by message privacy), so it must drop at some index i. By index hiding, such a drop can only occur if sk_i is known to the decoder, identifying a traitor.

1.1 Our Contribution

A Tracing Transform with Partial Decryptions. The current TPLBE definition does not account for partial decryptions and is therefore insufficient to achieve traceability if the decoder can request additional partial decryptions or if colluders have access to a partial decryption oracle when constructing a decoder.

Real partial decryptions may help a decoder distinguish which honest parties are revoked, thereby breaking index hiding. Indeed, the decoder can simply attempt decryption using the observed partial decryptions and conclude that a party is revoked if decryption is unsuccessful. To address this, we extend threshold PLBE with a SimPD algorithm which allows the tracer to generate *simulated partial decryptions* for ciphertexts. Crucially, simulated partial decryptions for parties that are revoked in the ciphertext

[1] As in the fingerprinting-based construction in [18], we consider *adaptive* partial insiders in this paper, allowing the decoder to learn partial decryptions from a set of up to $t-1$ parties *adaptively* selected for each ciphertext query.

should appear as if those parties were not revoked. Hence, the decoder can no longer distinguish the revocation status of honest parties based on the partial decryptions it observes.

To this end, we define the primitive *Threshold PLBE with Partial Decryptions* (TPLBE-PD) (see Section 4), extending TPLBE as outlined above.

With this primitive in hand, we present a compiler from TPLBE-PD to traceable threshold encryption with partial insiders (Section 5). This compiler extends the existing compilers of [20, 31], by simulating partial decryptions for the insiders requested by the decoder, making use of the new SimPD algorithm of TPLBE-PD. Importantly, the SimPD algorithm also enables us to simulate a partial decryption oracle for valid ciphertexts in the tracing game.

A TPLBE-PD Construction. In Section 6, we show how to instantiate our TPLBE-PD notion from pairings. Notably, our construction does not rely on a trusted dealer, nor does it require committee members to interact with each other for key generation,[2] while having a ciphertext size constant in the number of parties n. At a very high level, we revisit the framework of [20] and resolve its biggest limitation, ensuring that partial decryptions do not allow the adversary to distinguish whether an honest party is revoked for a certain ciphertext.

Strengthening the Partial Decryption Oracle. We initially present our constructions with respect to a partial decryption oracle *for valid ciphertexts* $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$, which requires the adversary to specify the randomness used for encryption when making oracle queries. However, we then show a generic compiler from traceability under $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ to traceability under a partial decryption oracle $\mathcal{O}^{\mathsf{PDec}}$ that works for arbitrary ciphertexts (Section 7). This contribution is of independent interest, as it not only allows our construction to be traceable under $\mathcal{O}^{\mathsf{PDec}}$, but also strengthens traceability guarantees achieved by constructions in [18], all of which were only analyzed under $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$. Roughly, our compiler works by attaching a proof of knowledge of the randomness used to generate a ciphertext and providing the tracer with a simulation trapdoor allowing it to generate arbitrary valid ciphertexts during tracing.

Putting it all together. Instantiating our compiler with our TPLBE-PD scheme leaves us with a Threshold Traitor Tracing scheme that is (a) resilient against up to $t-1$ adaptive partial insiders (maximal), (b) does not rely on a trusted dealer for key generation, (c) remains traceable even if the adversary has access to a partial decryption oracle, (d) has constant-size ciphertexts, and (e) can trace distinguishing decoders. We compare our result with the state of the art in Fig. 1.

Scheme	Insider Res.	Dist. Decoder	$\mathcal{O}^{\mathsf{PDec}}$	Dealer -Free	$\lvert\mathsf{ct}\rvert$	$\lvert\mathsf{pk}\rvert$	$\lvert\mathsf{sk}_i\rvert$
Naive	max. full	✓	✓†	✓	$O(n)$	$O(n)$	$O(1)$
Bormet et al. [20]	✗	✓	✗	✓	$O(1)$	$O(n^4)$	$O(1)$
Boneh et al. [11, §5.2]	✗	✓	✗	✗	$O(1)$	$O(1)$	$O(n^2)$
Boneh et al.[11, App. B]	✗	✓	✗	✗	$O(n^{1/3})$	$O(n^{1/3})$	$O(1)$
Bhattacharyya et al. [5]	✗	✓	✗	✗	$O(1)$	$O(1)$	$O(n^2)$
Chakrabarti et al. [24]	✗	✗	✗	✓	$O(\sqrt{n})$	$O(n)$	$O(1)$
Cannard et al. [23]	✗	✗	✗	✗	$O(n^2)$	$O(n^4)$	$O(n^2)$
Das et al. [31]	✗	✓	✗	✗	$O(1)$	$O(1)$	$O(1)$
Bormet et al. [18]	max. adapt. partial	✗	✓†	✗	$O(1)$	$O(1)$	$O(n^2)$
This work	max. adapt. partial	✓	✓	✓	$O(1)$	$O(n^4)$	$O(1)$

Figure 1: Comparison of TTT schemes. "Dist. Decoder" denotes whether the scheme supports distinguishing decoders or only decrypting decoders. Maximal insider resilience refers to resilience against $t-1$ insiders. Sublinear ciphertexts are considered a requirement to make threshold traitor tracing nontrivial. We note that the naive scheme, as well as [24] and [23] additionally achieve public traceability. †: proven only under partial decryption oracle for *valid ciphertexts* ($\mathcal{O}^{\mathsf{PDec\text{-}Val}}$), but our transform applies

[2]The construction requires a structured CRS. However, the same CRS can be reused across many setup executions.

1.2 Discussion and Future Work

We next discuss extensions and future research directions arising from our work.

Better efficiency. Whereas our construction has public key size $O(n^4)$, we note that only $O(n^3)$ of it is needed for encryption, while the rest is used solely for decryption. Reducing the public key size remains the most important open efficiency improvement for our construction.

Achieving Full Insiders. To the best of our knowledge, the only full-insider resilient constructions in the literature appear in [18], who present several compilers from any threshold traitor tracing scheme to a scheme that is resilient against full insiders via multi-traitor tracing. Crucially, if the underlying scheme is already (maximally) partial-insider resilient, the compiled scheme inherits a significant level of partial-insider resilience in addition to being resilient against full insiders, further motivating the study of purely partial-insider resilient constructions. As an example, instantiating the "unbalanced-bins" compiler in [18] with our construction yields a scheme that is resilient against $O(\sqrt{n})$ full insiders and additionally $n/3$ partial insiders for threshold $t = n/2$, at the cost of increasing ciphertext size to $O(\sqrt{n})$. Interestingly, the public key size *decreases* to $O(n^{2.5})$.

Other TPLBE-PD Constructions. We present a framework for constructing threshold traitor tracing against partial insiders. An interesting direction is to instantiate our framework with new constructions. A particularly interesting approach is to achieve TPLBE-PD from the construction of [31]. The resulting scheme would have optimal parameter sizes, but unfortunately would require a trusted dealer.

1.3 Additional Related Work

Traitor tracing has been extensively studied in the broadcast encryption setting, e.g., [29, 49, 39, 27, 15, 42, 40, 41, 38].

To the best of our knowledge, no previous works addressed partial insiders in threshold traitor tracing except [18]. However, [11] considers a weaker model which allows the decoder to receive additional partial decryptions, but assumes small coalitions and does not allow the decoder to choose which parties provide them. We emphasize that we consider decoders constructed by coalitions of arbitrary size, including *large coalitions*, i.e., coalitions of t or more colluders.

Aside from threshold encryption, traceability is also studied in related distributed primitives such as traceable secret sharing [12, 44, 36, 43] and traceable threshold VRFs [13]. However, in these primitives, the focus is on small coalitions.

A recent line of work [34, 17] has studied collusion countermeasures for secret sharing. Instead of modeling collusion via reconstruction boxes, they even consider that committee members may secretly reconstruct the secret in MPC. However, they rely on strong non-standard assumptions of MPC-hardness, and it is unclear how to apply these techniques to threshold encryption.

2 Technical Overview

We give a high-level overview of our techniques. For simplicity, we describe all constructions as *key encapsulation mechanisms* (KEM) with key space $\mathcal{K}$. Extensions to encryption follow via standard techniques.

2.1 The Tracing Definition

The first formal definition of threshold traitor tracing appears in [11], extending threshold encryption with a tracing algorithm Trace. In the corresponding security game, the challenger generates public key pk, secret key shares $\mathsf{sk}_1, \ldots, \mathsf{sk}_n$, and a tracing key tk. The adversary corrupts a set of traitors $\mathcal{T}$ of an *arbitrary* size via a corruption oracle $\mathcal{O}^{\mathsf{corrupt}}$, obtaining their secret keys, and outputs a decoder D. Then, the challenger runs the tracing algorithm $\mathcal{T}' \xleftarrow{\$} \mathsf{Trace}^{D}(\mathsf{pk}, \mathsf{tk})$ under black-box access to D and outputs the set of alleged traitors $\mathcal{T}'$.

A decoder D is *good* if it has a noticeable advantage in distinguishing whether a given KEM key k is encapsulated in a given ciphertext c, or uniformly random from the key space. Traceability requires: (a) tracing a good decoder outputs at least one real traitor ($\mathcal{T}' \neq \emptyset$ and $\mathcal{T}' \subseteq \mathcal{T}$), except with negligible probability; and (b) no honest party is framed ($\mathcal{T}' \subseteq \mathcal{T}$), except with negligible probability. Importantly, semantic security must hold, even if the tracer is malicious.

Traceability under Partial Decryption Oracle. In many practical scenarios, the colluders can observe partial decryptions of honest parties throughout protocol execution. These partial decryptions

should not allow colluders to evade tracing or frame honest parties. To model this, [18] introduces a partial decryption oracle $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ for *valid ciphertexts*. The adversary provides randomness $r_{\mathsf{Enc}} \in \mathcal{R}$, the oracle computes $c \leftarrow \mathsf{Enc}(\mathsf{pk}; r_{\mathsf{Enc}})$, and returns a partial decryption on c for selected party i. Henceforth, when we say *valid ciphertext*, we refer to ciphertexts that are computed honestly from (possibly adversarially chosen) randomness $r_{\mathsf{Enc}} \in \mathcal{R}$ as above. In this paper, we will present our constructions with respect to $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$, and then show a generic transform to traceability under a fully malicious partial decryption oracle $\mathcal{O}^{\mathsf{PDec}}$ that receives arbitrary ciphertexts in Section 7.

Adaptive Partial Insider Resilience. [18] extends the model to include partial insiders by letting the adversary specify a set $\mathcal{I}_{\mathsf{pd}} \subseteq [n]$ alongside the decoder D. To capture that the scheme must be able to trace decoders that receive partial decryptions from parties in $\mathcal{I}_{\mathsf{pd}}$, the definition of a *good decoder* is modified such that the decoder is additionally given the corresponding partial decryptions for the ciphertext c. We strengthen this further by allowing the decoder to choose $\mathcal{I}_{\mathsf{pd}}$ *adaptively per query*, i.e., after observing the query (k, c). The decoder can output any set $\mathcal{I}_{\mathsf{pd}} \subseteq [n]$ as long as $|\mathcal{I}_{\mathsf{pd}}| \leq \rho$, where the parameter ρ denotes the number of partial insiders tolerated by the scheme. A scheme is *maximally partial insider resilient* if $\rho = t - 1$.

Finally, we consider traceability without a trusted dealer, allowing the adversary to take control of traitors already during the Setup protocol.

2.2 TPLBE-PD: Threshold PLBE with Partial Decryptions

Our approach to achieving these strong traceability requirements is to define and construct a strengthened threshold PLBE notion called TPLBE-PD. We will next explore which properties are needed from the underlying TPLBE-PD scheme.

Assume a tracer gets access to a decoder D. Whenever the tracer queries D with some tuple (k, c), the decoder will adaptively output a set of partial insiders $\mathcal{I}_{\mathsf{pd}}$, and proceed to distinguish (i.e., decrypt) only after the tracer provides partial decryptions for c on behalf of parties in $\mathcal{I}_{\mathsf{pd}}$. To allow the tracer to simulate partial decryptions to the decoder for ciphertexts, where he knows the encryption randomness r_{Enc}, our TPLBE-PD notion provides a $\mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}}) \to d_i$ algorithm, where i refers to the target party.

Simulability of Partial Decryptions. Suppose an adversary might be able to construct a decoder that works perfectly when fed with *real* partial decryptions, but refuses to decrypt when fed with simulated partial decryptions during tracing. Such a decoder clearly breaks the scheme's security but would be impossible to trace. To address this attack, we require that simulated partial decryptions (stemming from SimPD) on valid ciphertexts are computationally indistinguishable from real partial decryptions (stemming from $\mathsf{PDec}(\mathsf{sk}_i, c)$), even if the adversary controls all secret key shares. We model this in a property called *simulability of partial decryptions*.

Indistinguishability with Simulated PDs. During tracing, the tracer first queries the decoder on $\mathsf{TrEnc}(0)$ ciphertexts. The tracing proof relies heavily on the fact that *good* decoders must also be good for $\mathsf{TrEnc}(0)$ ciphertexts, and thus threshold PLBE requires that normal (Enc) ciphertexts are indistinguishable from $\mathsf{TrEnc}(0)$ ciphertexts. For TPLBE-PD, we need to extend this requirement to ensure that Enc and $\mathsf{TrEnc}(0)$ remain indistinguishable, even if the decoder can ask for simulated partial decryptions.

Index Hiding with Simulated PDs. The tracing algorithm proceeds by progressively revoking parties until the decoder's success probability drops by a significant amount. If the decoder performs significantly worse on $\mathsf{TrEnc}(i^*)$ ciphertexts compared to $\mathsf{TrEnc}(i^* - 1)$, the tracer accuses party i^* as traitor. Recall that the standard PLBE argument relies on the intuition that a drop in the decoder's success probability for $\mathsf{TrEnc}(i^*)$ can only be explained by the adversary having access to sk_{i^*} when constructing the decoder. Now, however, we must consider that the decoder also receives simulated partial decryptions, possibly for party i^*.

Suppose, for now, that simulated partial decryptions were *identical* to real partial decryptions for all ciphertexts. This would mean that a simulated partial decryption for $\mathsf{TrEnc}(i^* - 1)$ is usable by the decoder to decrypt, as party i^* is not revoked, while a simulated partial decryption for $\mathsf{TrEnc}(i^*)$ is unusable, as party i^* is revoked. Hence, the decoder could distinguish $\mathsf{TrEnc}(i^*)$ and $\mathsf{TrEnc}(i^* - 1)$ without knowledge of sk_{i^*}, merely by checking whether the simulated partial decryption for i^* can be used to decrypt or not.

Intuitively, this means that simulated partial decryptions should be *valid*, in the sense that a simulated partial decryption for party i is consistent with the ciphertext as if i was not revoked, even if i is, in fact, revoked. Conceptually, this corresponds to "unrevoking" parties during the tracing procedure. While this appears to undermine the core PLBE mechanism at first glance, we ensure its structure is preserved by

requiring the decoder's success probability to drop *eventually* due to message privacy, as discussed below.

To capture this, we define *index hiding* such that decoders cannot meaningfully distinguish $\mathsf{TrEnc}(i^*)$ from $\mathsf{TrEnc}(i^* - 1)$, even when provided with simulated partial decryptions for parties including i^*. This then guarantees that when a drop in the success probability is observed at $\mathsf{TrEnc}(i^*)$, this must be because the decoder knows sk_{i^*}, and not because of simulated partial decryptions.

Message Privacy with Simulated PDs. Finally, we ensure that the success probability must drop eventually, even with simulated partial decryptions that behave as if parties are unrevoked. For this, we extend the message privacy property such that, given k and a ciphertext c generated from $\mathsf{TrEnc}(n)$, an adversary cannot distinguish whether k is encapsulated in c, even if it can adaptively query for up to $t-1$ simulated partial decryptions.

Last but not least, observe that we can simulate the $\mathcal{O}^{\mathsf{PDec\text{-}Val}}(i, r_{\mathsf{Enc}})$ oracle to adversaries running in the tracing game by just computing $c \leftarrow \mathsf{Enc}(\mathsf{pk}; r_{\mathsf{Enc}})$, and returning $d_i \leftarrow \mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$. Due to *simulability of partial decryptions*, this is indistinguishable from a real partial decryption for valid ciphertexts.

2.3 Starting Point: The Construction of BHO25 [20]

Our construction follows the TPLBE framework introduced by Bormet, Hofmann, and Othman [20]. Next, we provide an overview of their construction and discuss its limitations in achieving traceability in the presence of a partial decryption oracle. The main building block in [20] is a *silent* threshold encryption scheme.

Silent Setup. In a silent threshold encryption scheme, parties non-interactively sample individual key pairs $(\mathsf{pk}_i, \mathsf{sk}_i)$ based on a structured common reference string CRS. The important feature of silent setup is that for any subset of parties S, one can compute a public encryption key ek_S, and an aggregation key ak_S from the CRS and the public keys $\{\mathsf{pk}_i\}_{i \in S}$. One can then encrypt to S with a threshold t using ek_S, generating a *succinct* ciphertext c. If at least t parties from S have issued partial decryptions on c, one can recover the encrypted message using the partial decryptions and the aggregation key ak_S. The key observation in [20] is that silent setup essentially allows to encrypt to an arbitrary subset of parties, and thus can be used to realize threshold PLBE as follows.

Threshold PLBE from Silent Setup. [20] uses a silent setup scheme to revoke parties by excluding them from the set S during encryption. Hence, $\mathsf{TrEnc}(i)$ can be realized by encrypting to ek_{S_i} for the set $S_i = \{i+1, \ldots, n\}$. This effectively revokes the set of parties $\mathcal{R}_i = \{1, \ldots, i\}$[3] from decryption. However, silent setup schemes do not guarantee that the set S remains private, violating index hiding.

To fix this issue, [20] rerandomizes the public keys before encrypting, and thus ensures that the set of parties that are revoked does not leak from the ciphertext. However, since the public keys are now rerandomized, it is necessary to attach the consistently rerandomized aggregation key ak to allow for correct decryption. Unfortunately, ak is linear in n, violating the efficiency requirement of $o(n)$ ciphertext size.

Preprocessing to the rescue. To achieve constant ciphertext size, [20] proposes to *reuse* rerandomized $(\mathsf{ek}, \mathsf{ak})$ pairs that correspond to a set S, instead of attaching a new one to every ciphertext. In particular, in a preprocessing phase, they let the tracer generate a set of N preprocessing keys $\mathcal{U}_{\text{Norm}}$ for $S = [n]$ (i.e., keys that revoke no parties). Additionally, the tracer generates N preprocessing keys $\mathcal{U}_{\text{TR}}$ for the various revoked sets used during tracing. In more detail, for every $i \in [0, n]$, the tracer generates a set of $d = N/(n+1)$ preprocessing keys $\mathcal{U}_{\text{TR}}[i]$ that encrypt to the set S_i, revoking the set $\mathcal{R}_i$. Each of these preprocessing keys consists of a tuple $(\mathsf{ek}, \mathsf{ak})$. The tracer shuffles $\mathcal{U} \xleftarrow{\$} \mathsf{Shuffle}(\mathcal{U}_{\text{Norm}}, \mathcal{U}_{\text{TR}})$ and publishes all preprocessing keys $\mathcal{U}$, storing the secret mapping between preprocessing keys and corresponding revoked sets as the tracing key tk. Observe that because of the rerandomization, an adversary that does not know the secret key sk_i of a given party i cannot distinguish keys that revoke the set $\mathcal{R}_{i-1}$ from keys that revoke the set $\mathcal{R}_i$, even if the adversary knows all other secret keys.

Normal encryption (Enc). To encrypt, the encryptor chooses a random set of λ keys $(K_1, \ldots, K_\lambda)$ from the preprocessing $\mathcal{U}$ and encrypts under each of them, including the indices $(\mathsf{idx}_1, \ldots, \mathsf{idx}_\lambda)$ of the used keys in $\mathcal{U}$ within the ciphertext c. Note that just encrypting under a single random key is insufficient, as it will be from $\mathcal{U}_{\text{Norm}}$ only with probability $1/2$ and thus might revoke some parties. By encrypting to λ key per ciphertext, correctness is amplified to $1 - 2^\lambda$.

Trace-Encrypt ($\mathsf{TrEnc}(i)$). To revoke a set $\mathcal{R}_i$ during tracing, the tracer samples $(K_1, \ldots, K_\lambda)$ such that all preprocessing keys used in the ciphertext revoke at least $\mathcal{R}_i$. Note that it is insufficient to pick keys just from $\mathcal{U}_{\text{TR}}[i]$. To see this, suppose an adversary controls any party i, and consider indistinguishability:

[3] For ease of notation, denote the empty revoked set as $\mathcal{R}_0 = \emptyset$.

If a ciphertext is generated normally according to Enc, the adversary would expect that at least some of the keys revoke the party i with high probability (which it can check using the secret key sk_i). A tracing ciphertext $\mathsf{TrEnc}(0)$ using only keys from $\mathcal{U}_{\text{TR}}[0]$, however, would never revoke any party. As a result, an adversary could construct a decoder that works only for ciphertexts in which at least one key revokes party i, which is a good decoder that breaks indistinguishability. To address this, the tracer adjusts the distribution by which keys are sampled for $\mathsf{TrEnc}(i)$ such that $\mathsf{TrEnc}(0)$ chooses keys from $\mathcal{U}_{\text{TR}}[i]$ for $i \geq 1$ with the same distribution as random sampling during Enc would. Similarly, $\mathsf{TrEnc}(i)$ chooses keys from $\mathcal{U}_{\text{TR}}[i']$ for $i' > i$ with the same distribution as $\mathsf{TrEnc}(i-1)$ would, which ensures index hiding. For further details, we refer to Section 6.

Limitations of BHO25. During decryption, each party must send a partial decryption share for each component of the ciphertext that it can decrypt. The combiner then identifies a component in which all participating parties are not revoked, combines the partial decryptions for that component, and outputs the final decryption. Unfortunately, this means that partial decryptions reveal which components of a ciphertext a party can decrypt and, consequently, which keys revoke that party. Hence, their construction is not traceable in the presence of a partial decryption oracle (even just for valid ciphertexts).

2.4 Overview of Our TPLBE-PD Construction

Next, we present our approach to deal with the limitations discussed above.

Partial decryptions without leaking revocation information. To address this issue, our approach is as follows. Let $c = ((c_1, \mathsf{idx}_1), \ldots, (c_\lambda, \mathsf{idx}_\lambda))$ be a ciphertext, where each component c_j is encrypted under the preprocessing key at position idx_j in $\mathcal{U}$. We ensure that, for every party i, the partial decryption corresponding to i is identical across all components c_j in which i is not revoked. Hence, each party needs to send only a single partial decryption in the decryption process, hiding the component from which it originates.[4] As a result, the partial decryptions reveal no information about which keys revoke the party, as long as it is unrevoked in *at least one* preprocessing key in the ciphertext.

This, however, is not sufficient yet: Consider an adversary that controls party 2 and wants to break index hiding against the honest party 1. To do so, the adversary wants to find all keys in $\mathcal{U}_{\text{TR}}[1]$ and construct a decoder that refuses to decrypt whenever the ciphertext contains a large number of keys from $\mathcal{U}_{\text{TR}}[1]$. Observe that using the secret key sk_2, the adversary can identify all keys where party 2 is revoked, i.e., the set $\mathcal{U}_{\text{TR}}[\geq 2] = \bigcup_{2 \leq i \leq n} \mathcal{U}_{\text{TR}}[i]$, and also note that all of these keys must also revoke party 1. Now, the adversary can craft a ciphertext by picking $\lambda - 1$ keys from $\mathcal{U}_{\text{TR}}[\geq 2]$, and any random remaining key K that does not revoke party 2. The adversary then queries the partial decryption oracle $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ for party 1 on that ciphertext and observes the output: If the partial decryption is invalid (i.e., $\perp$), then the adversary knows that K revokes only party 1 but not party 2, and thus $K \in \mathcal{U}_{\text{TR}}[1]$. In this manner, the adversary can identify the set $\mathcal{U}_{\text{TR}}[1]$ step by step and construct a decoder that frames party 1.

This attack can be prevented by forcing the encryptor to select the keys used for encryption according to the output of a hash function $\mathcal{H}$ (modeled as a random oracle) as $((K_1, \mathsf{idx}_1), \ldots, (K_\lambda, \mathsf{idx}_\lambda)) \leftarrow \mathsf{Samp}(\mathcal{U}; \mathcal{H}(\mathsf{pre}))$ for a preimage pre that can be chosen by the encryptor at random.

Observe that this prevents the attack outlined above against a partial decryption oracle for valid ciphertexts: Since half of the preprocessing keys stem from $\mathcal{U}_{\text{Norm}}$ and thus revoke no parties, we can bound the probability of finding a preimage pre such that *all* λ selected keys revoke a party by $q \cdot 2^{-\lambda}$, where $q = \mathsf{poly}(\lambda)$ denotes the number of queries made to $\mathcal{H}$ by the adversary.

This can then be extended to support a fully malicious partial decryption oracle $\mathcal{O}^{\mathsf{PDec}}$, by requiring that encryptors attach a NIZK-PoK π proving knowledge of a preimage pre used in the encryption algorithm. To ensure that the tracer can still arbitrarily change the distribution of keys during TrEnc without the decoder noticing, the tracer would receive a trapdoor for the encryption proof system. We prove this transform from $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ to $\mathcal{O}^{\mathsf{PDec}}$ generically in Section 7. We discuss in Remark 7.5 how this transformation applies to our construction.

We next present a new construction that realizes the ideas outlined above. Our construction is based on the recent silent threshold encryption scheme by Waters and Wu [48], referred to as WW25.

Overview of WW25 [48]. The construction is based on a silent threshold Boneh–Boyen (BB) signature scheme introduced in the same paper. In particular, each ciphertext is encrypted under a tag τ, and decryption is possible only given a valid BB signature on τ; that is, the signature serves as a witness for decryption. However, extending the construction to TPLBE-PD introduces some challenges. Specifically, since each partial decryption corresponds to a partial signature on τ, a given partial decryption from party i can be checked, via the signature verification procedure, for consistency with a component c_j.

[4] Using sk_i, a party can efficiently check for which preprocessing keys it is unrevoked.

This leaks information about the components in which party i is revoked and those in which it is not. Thus, index-hiding is violated. In this paper, we introduce a variant of this construction that helps us to mitigate this issue.

A variant of WW25. Our key technical modification is to move decryption shares into the target group, preventing adversaries from testing the consistency of partial decryptions against ciphertext components, leading to the following modified construction.

- CRS: As in [48], let $s_1, \ldots, s_n$ be t-out-of-n Shamir shares of a random secret s, and let $c_i \xleftarrow{\$} \mathbb{Z}_p^*$ for all $i \in [n]$ be random cross terms. The CRS contains:

$$[s]_T, \{[c_i]_1, [c_i s_i]_1, [1/c_i]_2\}_{i\in[n]}, \{[s_i c_i/c_j]_1\}_{i\neq j\in[n]}$$

- KGen: Party i samples $\alpha_i \xleftarrow{\$} \mathbb{Z}_p^*$ and publishes: $[c_i\alpha_i]_1$ and $\{[\alpha_i c_i/c_j]_1\}_{j\in[n]\setminus\{i\}}$.

- Public encryption key: $\mathsf{ek} \leftarrow \sum_{i=1}^n [c_i(s_i+\alpha_i)]_1$.

- Public Aggregation key ak: Compute $\pi_i \leftarrow \sum_{j\neq i} \left[\frac{c_j}{c_i}\cdot(s_j+\alpha_j)\right]_1$ for all parties $i \in [n]$, and set $\mathsf{ak} \leftarrow \{\pi_i\}_{i\in[n]}$.

- Enc: To encapsulate a KEM key k under ek, sample $v \xleftarrow{\$} \mathbb{Z}_p$ at random and set $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2) = (v\cdot\mathsf{ek}, [v]_2)$. The corresponding KEM key is $k = [sv]_T$.

- The partial decryption of party i is:

$$d_i \leftarrow \frac{\mathsf{ct}_1 \circ [1/c_i]_2}{[\alpha_i]_1 \circ \mathsf{ct}_2 + \pi_i \circ [v]_2} = \frac{[v(s_i+\alpha_i)]_T + \sum_{j\neq i}\left[v\frac{c_j}{c_i}(s_j+\alpha_j)\right]_T}{[v\alpha_i]_T + \sum_{j\neq i}\left[v\frac{c_j}{c_i}(s_j+\alpha_j)\right]_T} = [s_i v]_T\,.$$

- Aggregation: Given partial decryption from a set S of at least t parties, interpolate $k \leftarrow \sum_{i\in S}\lambda_{i,S}^{(0)}\cdot[s_i v]_T = [sv]_T$

To gain intuition for semantic security, note that the decryption requires canceling the term $[\alpha_i v]_T$, which cannot be achieved without knowing $[\alpha_i]_1$, which is the secret key of party i. This is proved under a q-type assumption that can be justified in the GGM.

Revoking parties. Next, to build TrEnc, we need to revoke parties privately (i.e., ek and ak should not leak the set of revoked parties), which we achieve by rerandomization. Specifically, we let β, sampled uniformly at random, serve as the rerandomization factor for unrevoked parties, while rerandomizing revoked parties with independent randomness. That is, the tracer does the following to prepare a preprocessing key for a given set of revoked parties $\mathcal{R}$:

- Sample $\beta \xleftarrow{\$} \mathbb{Z}_p^*$. For all revoked parties $i \in \mathcal{R}$, sample $r_i \xleftarrow{\$} \mathbb{Z}_p^*$, and for all *unrevoked* parties $i \in [n]\setminus\mathcal{R}$ set $r_i \leftarrow \beta$.

- Compute the rerandomized encryption key $\mathsf{ek} \leftarrow \sum_{i=1}^n [c_i(s_i + r_i\alpha_i)]_1$.

- Compute the rerandomized aggregation key ak by computing for every party $i \in [n]$: $\pi_i = \sum_{j\neq i}[(s_j + r_j\alpha_j)c_j/c_i]_1$.

- Output the preprocessing key $([\beta]_1, \mathsf{ek}, \mathsf{ak} := \{\pi_i\}_{i\in[n]})$.

To encrypt under the preprocessing key, one computes $\mathsf{ct} = (v\cdot\mathsf{ek}, [v]_2)$. The partial decryption of the i-th party requires $[\alpha_i\beta]_1$ instead of $[\alpha_i]_1$, which can be computed from $[\beta]_1$ and the field element α_i. That is:

$$d_i \leftarrow \frac{v\cdot\mathsf{ek}\circ[1/c_i]_2}{[\alpha_i\beta]_1\circ[v]_2 + \pi_i\circ[v]_2} = [s_i v]_T\,.$$

To gain intuition for semantic security, it is easy to see that re-randomization does not give power to an adversary, even if it controls the field elements β and $\{r_i\}_{i\in\mathcal{R}}$. To see this, observe that, even knowing β, the tracer (who generates the preprocessing) cannot compute $[\alpha_i\beta v]_T$ from the public keys and ciphertext.[5]

[5]Here we implicitly require that the tracer attach a proof of knowledge of the randomness used in the preprocessing, certifying that he rerandomized properly.

To build intuition for why a revoked party cannot decrypt (which is particularly relevant for message privacy), one can view the revocation process as effectively transforming the party's key from $[\alpha_i]_1$ to $[\alpha_i r_i]_1$ for a random r_i. The only information that can be derived from the preprocessing key about r_i is of the form $[\alpha_i c_i r_i]_1$ and $[\alpha_i c_i r_i / c_j]_1$ for $i \neq j$. Observe that each party has published analogous information (corresponding to its own secret key α_i) as part of the public keys. Thus, from the perspective of a revoked party i, the value r_i plays the role of a "new" secret, about which it has no more information than it does about the secret keys of other parties. Therefore, this information does not help in decryption.

Furthermore, the tuple $([\beta]_1, \mathsf{ek}, \mathsf{ak})$ does not leak whether a party i is revoked due to rerandomization: Assuming hardness of the external Diffie-Hellman problem (XDH) [2, 7], an adversary given $[\beta]_1$ and $[r_i \alpha_i]_1$ cannot distinguish whether party i is rerandomized with $r_i = \beta$ (unrevoked), or with $r_i \xleftarrow{\$} \mathbb{Z}_p^*$ (revoked) without knowing α_i.

Encryption to multiple keys. Recall that following the framework of [20] (see Section 2.3), it is necessary to encrypt under multiple keys. To encrypt the same KEM key k to a set of keys $(K_1, \ldots, K_\lambda)$, where $K_q = ([\beta_q]_1, \mathsf{ek}_q, \mathsf{ak}_q)$ for $q \in [\lambda]$, we sample a single $v \xleftarrow{\$} \mathbb{Z}_p$, and encrypt using the same v under every K_q as outlined before. Let the resulting ciphertext be $c = (c_1, \ldots, c_\lambda)$, where the c_q component is an encryption under K_q (and randomness v). Observe that this now solves the partial decryption issue in [20]. Indeed, for every K_q it holds that if party i is unrevoked in K_q, then its partial decryption for c_q will be $[s_i v]_T$. Thus, the partial decryption no longer leaks from which c_q it comes, and hence does not reveal any information about which keys revoke party i. This also allows us to realize the SimPD algorithm: With knowledge of the randomness v, one can compute $[s_i v]_T$, since $[s_i]_T$ can be computed from the CRS. This is identical to real partial decryptions, unless the party i is revoked in *all* preprocessing keys used in the ciphertext. Recall that generating such ciphertexts is infeasible, since preprocessing keys are selected according to a random oracle.

Applying this idea naively leads to issues with semantic security. Recall that our notion of semantic security must also hold against the tracer, who is aware of all β-values used during rerandomization. Hence, reusing the same randomness v across ciphertext components may introduce correlations that must be carefully controlled to avoid leaking information. For example, suppose the ciphertext includes terms $v \cdot \mathsf{ek}_1$ and $v \cdot \mathsf{ek}_2$, where $\mathsf{ek}_1 = [c_1 s_1 + c_1 \beta_1 \alpha_1]_1$ and $\mathsf{ek}_2 = [c_1 s_1 + c_1 \beta_2 \alpha_1]_1$ (considering a single party for simplicity). Then the tracer can compute $(v \cdot \mathsf{ek}_1) - (v \cdot \mathsf{ek}_2) = [v c_1 \alpha_1 (\beta_1 - \beta_2)]_1$. Since the tracer knows β_1 and β_2, it can easily derive $[c_1 \alpha_1 v]_1$. From this, by pairing with $[1/c_1]_2$, it can compute $[\alpha_1 v]_T$, which, according to the decryption procedure, suffices to decrypt given β as a field element.

Breaking dependency between components. To address this issue, we let each party sample a secret key $\alpha_i^{(q)}$ for each of the $q \in [\lambda]$ positions in a ciphertext. For each preprocessing key, we then generate λ values $(\mathsf{ek}_1, \ldots, \mathsf{ek}_\lambda)$ and $(\mathsf{ak}_1, \ldots \mathsf{ak}_\lambda)$ using the same preprocessing randomness β and r_i, where the q'th component is constructed with the parties' q'th public keys. When a preprocessing key is used in the q'th position of a ciphertext, we use its q'th component during encryption. This resolves the issue described above, since now ek_1 depends on $\alpha_1^{(1)}$ while ek_2 depends on $\alpha_1^{(2)}$. That is, we have $[v \cdot \mathsf{ek}_1 - v \cdot \mathsf{ek}_2]_1 = \left[v c_1 (\beta_1 \alpha_1^{(1)} - \beta_2 \alpha_1^{(2)})\right]_1$. The intuition is that such expressions involve linear combinations of independent secret keys, and therefore do not allow the adversary to isolate any term corresponding to a valid partial decryption share, without knowing any of the elements $\{\alpha_i^{(q)}\}_{q \in [\lambda]}$. We provide a formal security proof in Section 6.1. Importantly, note that the partial decryptions are still $[s_i v]_T$.

3 Preliminaries

Notation. We denote the security parameter by λ, and write $\mathsf{negl}(\lambda)$ (resp. $\mathsf{poly}(\lambda)$) for functions that are negligible (resp. polynomials) in λ. We denote the number of parties by n. When describing asymptotic properties of our constructions, we ignore factors other than n in the O-notation. For integers $a, b \in \mathbb{N}$, we denote the set $\{1, \ldots, b\}$ by $[b]$ and the set $\{a, \ldots, b\}$ by $[a, b]$. We write $y \leftarrow A(x; r)$ to execute the algorithm A on input x with randomness r and assign the result to y. If A is executed with uniform randomness, we abbreviate $y \xleftarrow{\$} A(x)$ to indicate that A is probabilistic. For an interactive protocol P among a set of parties S we use the notation $(y) \xleftarrow{\$} P(x) \left[\mathcal{A}^{\mathcal{O}} : S'\right]$ within security games, to denote that the challenger executes protocol P on input x such that the adversary $\mathcal{A}$ has access to the oracle $\mathcal{O}$ and controls the parties $S' \subseteq S$, and the challenger controls the remaining parties $S \setminus S'$. The specifics of public and private inputs for different parties are defined by the protocol P.

Sampling without replacement. To sample m values without replacement from an indexed set S, we write $((a_1, \mathsf{idx}_1), \ldots (a_m, \mathsf{idx}_m)) \xleftarrow{\$} \mathsf{Samp}(S, m)$, where $a_i \in S$, and idx_i is the index of a_i in S. We

denote the randomness space of $\mathsf{Samp}(S, m)$ as $\mathcal{S}_{S,m}$.

Lagrange Interpolation. Denote with $\lambda_{i,S}^{(x)} = \prod_{j \in S \setminus \{i\}} \frac{x-j}{i-j}$ the Lagrange coefficient for interpolating a polynomial at position x using index i among the set of indices S. Then, given points (i, y_i) for $i \in S$, the polynomial $p(x) = \sum_{i \in S} \lambda_{i,S}^{(x)} \cdot y_i$ is of degree $|S| - 1$, and $p(i) = y_i$ for all $i \in S$.

Commitments. We use a commitment scheme $(\mathsf{Setup}, \mathsf{Com})$. For a message m and randomness r, the $\mathsf{Com}(m; r)$ algorithm produces a commitment com. We require standard computational hiding and binding (c.f. Appendix A).

Pairing Groups. A bilinear pairing ensemble $E = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \circ, [1]_1, [1]_2, p)$ consists of cyclic groups $\mathbb{G}_1$ with generator $[1]_1$, $\mathbb{G}_2$ with generator $[1]_2$, and $\mathbb{G}_T$ of prime order p with a pairing operation $\circ\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ that satisfies bilinearity and non-degeneracy. We usually write $\circ$ infix.

We rely on the external Diffie-Hellman assumption [2, 7].

Assumption 1 (External Diffie Hellman (XDH)). Let $\mathcal{G}$ be an algorithm to generate a pairing ensemble E. The extenal Diffie Hellman problem (XDH) is hard for $\mathcal{G}$ if for all PPT adversaries $\mathcal{A}$, the following function is negligible:

$$\mathsf{Adv}^{\mathsf{XDH}}_{\mathcal{A},\mathcal{G}}(1^\lambda) := \left|\Pr\left[\mathcal{A}(1^\lambda, E, [x]_1, [y]_1, [xy]_1) = 1\right] - \Pr\left[\mathcal{A}(1^\lambda, E, [x]_1, [y]_1, [z]_1) = 1\right]\right|,$$

where $E = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \circ, [1]_1, [1]_2, p) \xleftarrow{\$} \mathcal{G}$ and $x, y, z \xleftarrow{\$} \mathbb{Z}_p^*$.

We also rely on the *n-extended bilinear decisional Diffie-Hellman assumption* (n-EBDDH), which is a generalization of the standard bilinear decisional Diffie-Hellman assumption [45, 9]. It was first introduced and proven to hold in the generic bilinear group model [47, 6] in [48].

Assumption 2 (n-Extended Bilinear Decisional Diffie-Hellman (n-EBDDH)). Let $\mathcal{G}$ be an algorithm to generate a pairing ensemble E, and let $n \in \mathsf{poly}(\lambda)$. The n-extended bilinear decisional Diffie Hellman problem (n-EBDDH) is hard for $\mathcal{G}$ if for all PPT adversaries $\mathcal{A}$, the following function is negligible in λ:

$$\mathsf{Adv}^{n\text{-}\mathsf{EBDDH}}_{\mathcal{A},\mathcal{G}}(1^\lambda) := \left|\Pr\left[\mathcal{A}(1^\lambda, \mathsf{view}, [abv]_T) = 1\right] - \Pr\left[\mathcal{A}(1^\lambda, \mathsf{view}, [z]_T) = 1\right]\right|,$$

where $E = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \circ, [1]_1, [1]_2, p) \xleftarrow{\$} \mathcal{G}$, $a, b, v, z \xleftarrow{\$} \mathbb{Z}_p$, $c_1, \ldots, c_n \xleftarrow{\$} \mathbb{Z}_p^*$, and $\mathsf{view} :=$

$$\left(E, [a]_1, [b]_2, [v]_2, \{[c_i]_1, [abc_i]_1, [vc_i]_1, [1/c_i]_2\}_{i \in [n]}, \{[c_i/c_j]_1, [abc_i/c_j]_1\}_{i \neq j \in [n]}\right).$$

3.1 Insider-Resilient Threshold Traitor Tracing

We recall the definition threshold traitor tracing due to [18], with minor adjustments to accommodate interactive setup protocols and adaptive partial insiders.

Definition 3.1 (Traceable Threshold KEM ($\mathsf{TT\text{-}KEM}$)). A traceable threshold KEM ($\mathsf{TT\text{-}KEM}$) with key space $\mathcal{K}_\lambda$ is a tuple of protocols and algorithms $\mathsf{TT\text{-}KEM} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{PDec}, \mathsf{Comb}, \mathsf{Trace})$ with the following syntax:

- $(\mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{i \in [n]}, \mathsf{tk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, n, t, \varepsilon)$. Setup is a (possibly) interactive protocol between a trusted party $\mathbb{D}$, n committee members $\{\mathbb{P}_i\}_{i \in [n]}$, and a tracer $\mathbb{T}$. All parties receive the security parameter 1^λ, the number of committee members n, the threshold t, and the limit ε as input. At the end of the protocol, each party $i \in [n]$ receives sk_i as private output, and a tracing key tk is private output for the tracer. The public parameters pp and public key pk are public outputs. Subsequent algorithms get pp as implicit input.
- $(k, c) \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk})$. Enc is a probabilistic algorithm that, given a public key pk, outputs a ciphertext c and an encapsulated key k.
- $d_i \leftarrow \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)$. Given public key pk, a secret key share sk_i on behalf of party $i \in [n]$, the PDec algorithm outputs a partial decryption d_i.
- $k \xleftarrow{\$} \mathsf{Comb}(\mathsf{pk}, c, S, \{d_i\}_{i \in S})$. Given the public key pk, a ciphertext c, a set of partial decryptions $\{d_i\}_{i \in S}$ on behalf of parties in $S \subseteq [n]$, the Comb algorithm outputs the encapsulated key k.
- $\mathcal{T}' \xleftarrow{\$} \mathsf{Trace}^{D(\cdot)}(\mathsf{pk}, \mathsf{tk}, \varepsilon)$. The tracing algorithm receives the public key pk, the tracing key tk, the limit ε, and black-box access to a decoder algorithm D. It outputs a set of traitors $\mathcal{T}'$.

$\mathsf{Game\text{-}Tr}_{\mathcal{A}}(1^\lambda, \varepsilon, \rho)$

$(1^n, 1^t, \mathsf{st}) \xleftarrow{\$} \mathcal{A}(1^\lambda) \quad /\!/\ 0 < t \leq n$

$\mathcal{T} \leftarrow \emptyset$

$(\mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{tk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, n, t, \varepsilon) \left[\mathcal{A}^{\mathcal{O}^{\mathsf{corrupt}}} : \{\mathbb{P}_i\}_{i\in\mathcal{T}}\right]$

$D \xleftarrow{\$} \mathcal{A}^{\mathcal{O}^{\mathsf{corrupt}}, \mathcal{O}^{\mathsf{PDec\text{-}Val}}}(\mathsf{st}, \mathsf{pk})$

$\mathcal{T}' \xleftarrow{\$} \mathsf{Trace}^D(\mathsf{pk}, \mathsf{tk}, \varepsilon)$

return $(n, t, D, \mathcal{T}, \mathcal{T}', \mathsf{pk}, \{\mathsf{sk}_i\}_{i\in[n]})$

$\mathcal{O}^{\mathsf{corrupt}}(i)$

if $i \notin [n]$ **return** $\perp$

Update $\mathcal{T} \leftarrow \mathcal{T} \cup \{i\}$

return sk_i

$\mathcal{O}^{\mathsf{PDec\text{-}Val}}(i, r_{\mathsf{Enc}})$

if $r_{\mathsf{Enc}} \notin \mathcal{R}_{\mathsf{Enc}}$ **return** $\perp$

$(k, c) \leftarrow \mathsf{Enc}(\mathsf{pk}; r_{\mathsf{Enc}})$

return $d_i \leftarrow \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)$

Figure 2: Traceability game for TT-KEM.

We strengthen the traceability definition of [18] to consider *adaptive partial insiders*. This means that, instead of fixing the set of partial insiders in advance, the decoder can adaptively request any set of partial insiders for each query. The definition in [18] also models multi-traitor tracing and full-insider resilience, which we do not consider in this paper.

Definition 3.2 (Traceability against ρ Adaptive Partial Insiders with limit ε). Let $\varepsilon(\lambda) \in (0, 1/2]$ and $\rho(\lambda, n, t)$ be functions in the security parameter. A TT-KEM scheme is traceable against ρ adaptive partial insiders, if for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$

$$\Pr[\mathsf{GoodTr}] \geq \Pr[\mathsf{GoodDec}] - \mathsf{negl}(\lambda) \quad \text{and} \quad \Pr[\mathsf{BadTr}] \leq \mathsf{negl}(\lambda),$$

where $(n, t, D, \mathcal{T}, \mathcal{T}', \mathsf{pk}, \{\mathsf{sk}_i\}_{i\in[n]}) \xleftarrow{\$} \mathsf{Game\text{-}Tr}_{\mathcal{A}}(1^\lambda, \varepsilon, \rho)$ (Fig. 2), and the three events GoodTr, BadTr and GoodDec are defined as follows:

- GoodTr occurs if $\mathcal{T}' \neq \emptyset$ and $\mathcal{T}' \subseteq \mathcal{T}$, which indicates that at least one real traitor was traced and no non-traitors were framed by the tracing algorithm.
- BadTr occurs if at least one non-traitor was framed ($\mathcal{T}' \neq \emptyset$ and $\mathcal{T}' \not\subseteq \mathcal{T}$).
- GoodDec indicates that the decoder is good for pk, when given partial decryptions on behalf of at most ρ parties per query. More formally, when $P(D) \geq 1/2 + \varepsilon(\lambda)$, where

$$P(D) \coloneqq \Pr\left[D(\mathsf{st}, \{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}}) = b : \begin{array}{c} (c, k_0) \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}); k_1 \xleftarrow{\$} \mathcal{K}_\lambda; b \xleftarrow{\$} \{0,1\} \\ (\mathcal{I}_{\mathsf{pd}}, \mathsf{st}) \xleftarrow{\$} D(k_b, c) \\ /\!/\ \mathcal{I}_{\mathsf{pd}} \subseteq [n] \wedge |\mathcal{I}_{\mathsf{pd}}| \leq \rho(\lambda, n, t) \\ \forall i \in \mathcal{I}_{\mathsf{pd}} \colon d_i \leftarrow \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c) \end{array} \right].$$

We say that as TT-KEM is maximally partial insider resilient, if it is traceable against $\rho(\lambda, n, t) = t - 1$ partial insiders.

We require that a TT-KEM is *correct* and achieves strong semantic security.

Definition 3.3 (Correctness of TT-KEM). A TT-KEM is correct if there exists a negligible function negl such that for all $\lambda, n, t \in \mathbb{N}$ with $0 < t \leq n$ and all $S \subseteq [n]$ with $|S| \geq t$ it holds that

$$\Pr\left[\begin{array}{c} \mathsf{Comb}(\mathsf{pk}, c, S, \{d_i\}_S) \\ = k \end{array} : \begin{array}{c} (\mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{[n]}, \mathsf{tk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, n, t) \\ (k, c) \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}) \\ \forall i \in S \colon d_i \leftarrow \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c) \end{array} \right] \geq 1 - \mathsf{negl}(\lambda).$$

Strong CPA Security (SIND-CPA). At a high level, (strong) semantic security says that an adversary corrupting at most $t - 1$ parties cannot distinguish whether a candidate KEM key k is encapsulated in a challenge ciphertext c, even if the adversary also corrupts the tracer. Following [31], we work with a stronger notion, where the adversary additionally has access to an $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ oracle, which allows the adversary to query for partial decryption of *valid ciphertexts* of its choice. This remains weaker than IND-CCA security, since ciphertexts cannot be chosen fully maliciously. For simplicity, we consider *static corruptions*. Unlike [31], we allow the adversary to control corrupted parties and the tracer already during the setup protocol, i.e., we do not rely on a trusted dealer.

Game $\mathsf{SIND\text{-}CPA}_{\mathcal{A}}(1^\lambda)$	Oracle $\mathcal{O}^{\mathsf{PDec\text{-}Val}}(i, r_{\mathsf{Enc}})$
$(1^n, 1^t, \mathsf{Cor}, \mathsf{st}) \xleftarrow{\$} \mathcal{A}(1^\lambda)$ // $0 < t \le n$, $\mathsf{Cor} \subseteq [n]$, $\lvert\mathsf{Cor}\rvert < t$	**if** $r_{\mathsf{Enc}} \notin \mathcal{R}_{\mathsf{Enc}}$ **return** $\perp$
$(\mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{tk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, n, t)\ \big[\mathcal{A}\colon \{\mathbb{P}_i\}_{i\in\mathsf{Cor}}, \mathbb{T}\big]$	$(k, c) \leftarrow \mathsf{Enc}(\mathsf{pk}; r_{\mathsf{Enc}})$
$(k_0, c) \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk})$	$d_i \leftarrow \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)$
$k_1 \xleftarrow{\$} \mathcal{K}$	**return** d_i
$b \xleftarrow{\$} \{0,1\}$	
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}^{\mathsf{PDec\text{-}Val}}}(\mathsf{st}, \mathsf{pk}, \{\mathsf{sk}_i\}_{i\in\mathsf{Cor}}, \mathsf{tk}, c, k_b)$	
return $b \stackrel{?}{=} b'$	

Figure 3: The SIND-CPA game of TT-KEM and TPLBE-PD.

Definition 3.4 (Strong Semantic Security (SIND-CPA)). A TT-KEM scheme satisfies strong semantic security if for all PPT adversaries $\mathcal{A}$, the advantage

$$\mathsf{Adv}^{\mathsf{SIND\text{-}CPA}}_{\mathcal{A},\mathsf{TT\text{-}KEM}}(\lambda) \coloneqq \Pr\big[\mathsf{SIND\text{-}CPA}_{\mathcal{A}}(1^\lambda) = 1\big] - \frac{1}{2}$$

is negligible in λ, where SIND-CPA is defined in Fig. 3.

4 Threshold PLBE with Partial Decryptions

We present our new notion of threshold PLBE with partial insiders (TPLBE-PD).

Definition 4.1 (Threshold PLBE with Partial Decryptions (TPLBE-PD)). A threshold PLBE with Partial Decryptions (TPLBE-PD) with key space $\mathcal{K}$ and encryption randomness space $\mathcal{R}$ is a tuple of algorithms and protocols $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{TrEnc}, \mathsf{PDec}, \mathsf{SimPD}, \mathsf{Comb})$. Its syntax is the same as that of TT-KEM, except that it has the following two algorithms instead of a Trace algorithm:

- $(k, c) \xleftarrow{\$} \mathsf{TrEnc}(\mathsf{tk}, j)$. Given the tracing key tk and an index $j \in [0, n]$, the TrEnc algorithm outputs a ciphertext c and an encapsulated key k.
- $d_i \leftarrow \mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$. Given the public key pk, an index $i \in [n]$, a ciphertext c, and encryption randomness r_{Enc}, the SimPD algorithm outputs a simulated partial decryption d_i.

We give a comprehensive overview of all algorithms in Appendix B.

Our TPLBE-PD must satisfy the standard correctness notion for PLBE schemes, requiring that t or more *unrevoked* parties can decrypt ciphertexts generated by Enc and TrEnc with overwhelming probability (c.f. Appendix B).

For convenience, we define the deterministic algorithm $\mathsf{SimPDs}(\mathcal{I}_{\mathsf{pd}}, \rho, \mathsf{pk}, c_b, r_{\mathsf{Enc}})$ which returns all simulated partial decryptions on behalf of $\mathcal{I}_{\mathsf{pd}}$ if $|\mathcal{I}_{\mathsf{pd}}| \le \rho$. More formally, if $|\mathcal{I}_{\mathsf{pd}}| > \rho$, output $\{\perp\}_{\mathcal{I}_{\mathsf{pd}}}$. Otherwise, for all $i \in \mathcal{I}_{\mathsf{pd}}$ compute $d_i \leftarrow \mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$ and output $\{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}}$.

$\mathsf{GameSimPD}_{\mathcal{A}}(1^\lambda)$	Oracle $\mathsf{Chal}(b, r_{\mathsf{Enc}})$
$(1^n, 1^t, \mathsf{st}) \xleftarrow{\$} \mathcal{A}(1^\lambda); b \xleftarrow{\$} \{0,1\}$	// Parse $r_{\mathsf{Enc}} \in \mathcal{R}$
$(\mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{tk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, n, t)\ \big[\mathcal{A}\colon \{\mathbb{P}_i\}_{i\in[n]}\big]$	$(k, c) \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}; r_{\mathsf{Enc}})$
$b' \xleftarrow{\$} \mathcal{A}^{\mathsf{Chal}(b,\cdot)}(\mathsf{st}, \mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{i\in[n]})$	**if** $b = 0$ **then**
return $b \stackrel{?}{=} b'$	**return** $\{d_i \leftarrow \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)\}_{i\in[n]}$
	return $\{d_i \leftarrow \mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})\}_{i\in[n]}$

Figure 4: Simulability of Partial Decryptions GameSimPD

We require that simulated partial decryptions on *valid ciphertexts* are computationally indistinguishable from real partial decryptions, even if the adversary knows all secret keys and can choose arbitrary random coins from $\mathcal{R}$ to encrypt.

Definition 4.2 (Simulability of Partial Decryptions). A TPLBE-PD has simulable partial decryptions if for all PPT adversaries $\mathcal{A}$, it holds that the advantage

$$\mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{A},\mathsf{TPLBE\text{-}PD}}(\lambda) \coloneqq \left|\frac{1}{2} - \Pr[\mathsf{GameSimPD}_{\mathcal{A}}(1^\lambda) = 1]\right|$$

is negligible in λ, where GameSimPD is defined in Fig. 4.

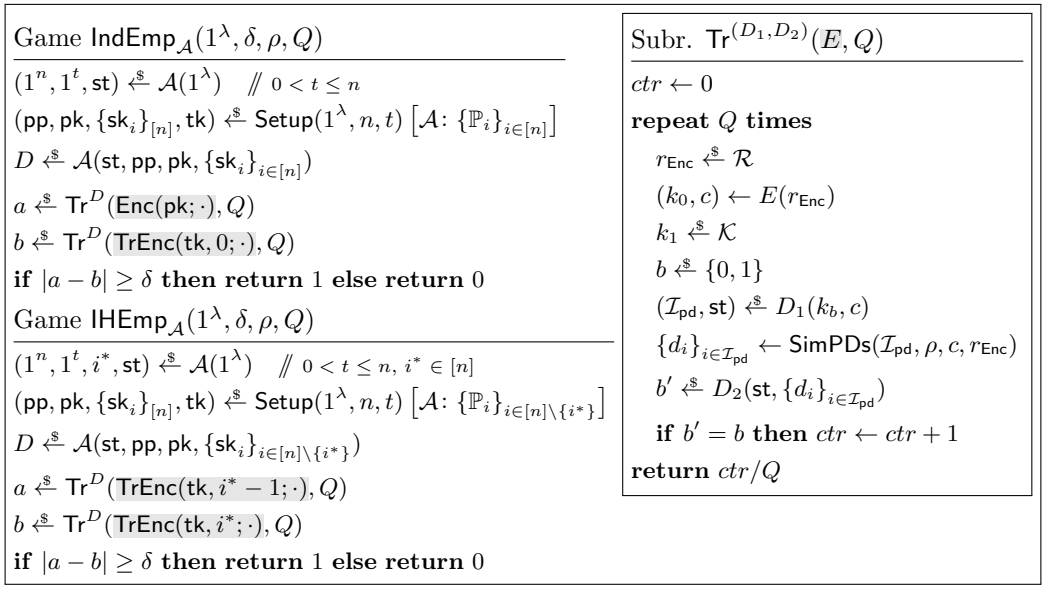

Figure 5: Empirical indistinguishability and index hiding games of TPLBE-PD

Indistinguishability and Index Hiding. We define *empirical variants* of the decoder-based indistinguishability and index hiding properties first introduced in [20]. Informally, indistinguishability guarantees that an adversary who controls all parties cannot produce a decoder that distinguishes between Enc and TrEnc(0) ciphertexts with advantage exceeding some threshold $\delta \in [0, \frac{1}{2}]$. Index hiding guarantees that an adversary who corrupts all parties except i^* cannot produce a decoder that distinguishes between $\mathsf{TrEnc}(i^*-1)$ and $\mathsf{TrEnc}(i^*)$ with advantage exceeding δ. As in [20], we allow δ to be bounded but non-negligible, and only require that any PPT adversary has negligible probability to produce a decoder whose advantage exceeds δ. [20] first showed that such relaxed indistinguishability and index hiding notions are already sufficient to prove traceability.

Additionally, we introduce the parameter $\rho < t$, and allow the decoder to adaptively request up to ρ simulated partial decryptions for every query.

Finally, our security games *empirically estimate* the decoder's distinguishing advantage across $Q \in \mathsf{poly}(\lambda)$ repetitions (see subroutine Tr in Fig. 5). This addresses a subtle oversight in the purely decoder-based definition due to [20]: There, the adversary's success is defined in terms of the decoder's exact distinguishing advantage, which is generally not efficiently computable in a black-box manner. As a result, their definitions are non-falsifiable and can complicate reduction-based proofs that need to determine whether an internal adversary wins the game to answer a challenge. To avoid such issues, we instead work with falsifiable empirical variants. Importantly, in the proof of our traceability compiler, we show that these relaxed notions remain sufficient.

Definition 4.3 ((δ, ρ, Q)-Indistinguishability). We say that a TPLBE-PD scheme achieves (δ, ρ, Q)-indistinguishability for $\delta \in [0, \frac{1}{2}]$, a function $\rho(\lambda, n, t)$, and $Q \in \mathsf{poly}(\lambda)$, if for all PPT adversaries $\mathcal{A}$, the advantage

$$\mathsf{Adv}^{\mathsf{IndEmp}}_{\mathcal{A},\mathsf{TPLBE\text{-}PD}}(\lambda, \delta, \rho, Q) \coloneqq \Pr\big[\mathsf{IndEmp}_{\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\big]$$

is negligible in λ, where IndEmp is defined in Fig. 5.

Definition 4.4 ((δ, ρ, Q)-Index Hiding). A TPLBE-PD scheme is (δ, ρ, Q)-index hiding for $\delta \in [0, \frac{1}{2}]$, a function $\rho(\lambda, n, t)$, and $Q \in \mathsf{poly}(\lambda)$, if for all PPT adversaries $\mathcal{A}$, the advantage

$$\mathsf{Adv}^{\mathsf{IHEmp}}_{\mathcal{A},\mathsf{TPLBE\text{-}PD}}(\lambda, \delta, \rho, Q) \coloneqq \Pr\big[\mathsf{IHEmp}_{\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\big]$$

is negligible in λ, where IHEmp is defined in Fig. 5.

Game $\mathsf{MP}_{\mathcal{A}}(1^\lambda, \rho)$

$(1^n, 1^t, \mathsf{st}) \xleftarrow{\$} \mathcal{A}(1^\lambda) \quad /\!/\ 0 < t \le n$

$(\mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{tk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, n, t)\ \big[\mathcal{A}\colon \{\mathbb{P}_i\}_{i\in[n]}\big]$

$b \xleftarrow{\$} \{0,1\}; \quad r_{\mathsf{Enc}} \xleftarrow{\$} \mathcal{R}$

$(k_0, c) \xleftarrow{\$} \mathsf{TrEnc}(\mathsf{tk}, n; r_{\mathsf{Enc}}); \quad k_1 \xleftarrow{\$} \mathcal{K}$

$(\mathcal{I}_{\mathsf{pd}}, \mathsf{st}) \xleftarrow{\$} \mathcal{A}^{\mathsf{TrEnc}(\mathsf{tk},\cdot,\cdot)}(\mathsf{st}, \mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{i\in[n]}, c, k_b)$

if $\mathcal{I}_{\mathsf{pd}} \not\subseteq [n] \vee |\mathcal{I}_{\mathsf{pd}}| > \rho$ **then return** 0

$\{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}} \leftarrow \mathsf{SimPDs}(\mathcal{I}_{\mathsf{pd}}, \rho(\lambda, n, t), \mathsf{pk}, c, r_{\mathsf{Enc}})$

$b' \xleftarrow{\$} \mathcal{A}(\mathsf{st}, \{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}})$

return $b \stackrel{?}{=} b'$

Figure 6: The message privacy game of TPLBE-PD

Message Privacy. Our message privacy definition captures that an adversary cannot have a significant advantage to distinguish whether a given KEM key k is random, or encapsulated in a ciphertext c that was generated using $\mathsf{TrEnc}(\mathsf{tk}, n)$ (i.e., that revokes all parties). This holds even if the adversary controls *all* committee members. Additionally, the adversary is granted access to a TrEnc oracle for arbitrary indices, and the adversary is allowed to adaptively query for up to ρ simulated partial decryptions on the challenge ciphertext.

Definition 4.5 (ρ-Message Privacy). A TPLBE-PD scheme is ρ-message private for $\rho(\lambda, n, t)$ if for all PPT adversaries $\mathcal{A}$, for advantage

$$\mathsf{Adv}^{\mathsf{MP}}_{\mathcal{A},\mathsf{TPLBE\text{-}PD}}(\lambda, \rho) \coloneqq \Pr\big[\mathsf{MP}_{\mathcal{A}}(1^\lambda, \rho) = 1\big] - \frac{1}{2}$$

is negligible in λ, where MP is defined in Fig. 6

We define strong semantic security identically to that of TT-KEM.

Definition 4.6 (SIND-CPA Security for TPLBE-PD). We say that a TPLBE-PD scheme satisfies strong semantic security if for all PPT adversaries $\mathcal{A}$, the advantage

$$\mathsf{Adv}^{\mathsf{SIND\text{-}CPA}}_{\mathcal{A},\mathsf{TPLBE\text{-}PD}}(\lambda) \coloneqq \Pr\big[\mathsf{SIND\text{-}CPA}_{\mathcal{A}}(1^\lambda) = 1\big] - \frac{1}{2}$$

is negligible in λ, where SIND-CPA is defined in Fig. 3.

5 Compiler: TPLBE-PD to Threshold Traitor Tracing

Algorithm $\mathsf{Trace}^D(\mathsf{pk}, \mathsf{tk}, \varepsilon)$

$Q \leftarrow 2^7 \cdot 3n^2\lambda/\varepsilon^2$

$\mathcal{T}' \leftarrow \emptyset$

for $i \in [0, n]$ **do**

  $ctr_i \leftarrow 0$

  repeat Q **times**

    $ctr_i \leftarrow ctr_i + \mathsf{TestTrEnc}(i)$

  $\hat{p}_i \leftarrow ctr_i/Q$

return $\mathcal{T}' \leftarrow \left\{i \in [n]\colon \hat{p}_{i-1} - \hat{p}_i \ge \frac{\varepsilon}{4n}\right\}$

Subroutine $\mathsf{TestTrEnc}^D(i)$

$b \xleftarrow{\$} \{0,1\}$

$((k_0, c); r_{\mathsf{Enc}}) \xleftarrow{\$} \Pi.\mathsf{TrEnc}(\mathsf{tk}, i)$

$(\mathcal{I}_{\mathsf{pd}}, \mathsf{st}) \xleftarrow{\$} D_1(k_b, c)$

if $\mathcal{I}_{\mathsf{pd}} \subseteq [n] \wedge |\mathcal{I}_{\mathsf{pd}}| \le \rho$ **then**

  $\{d_j \leftarrow \Pi.\mathsf{SimPD}(\mathsf{pk}, j, c, r_{\mathsf{Enc}})\}_{j\in\mathcal{I}_{\mathsf{pd}}}$

else $\{d_j \leftarrow \bot\}_{j\in\mathcal{I}_{\mathsf{pd}}}$

if $D_2(\mathsf{st}, \{d_j\}_{j\in\mathcal{I}_{\mathsf{pd}}}) = b$ **return** 1

else return 0

Figure 7: The Trace algorithm with respect to a TPLBE-PD scheme Π

Construction 1. Let Π be a TPLBE-PD scheme. Then, we construct a TT-KEM Π' as follows. The Setup, Enc, PDec, and Comb algorithms of Π' are identical to those of Π, and the Trace algorithm is defined in Fig. 7.

Theorem 5.1 *Let* Π *be a* TPLBE-PD *scheme with* $(\frac{\varepsilon}{8}, \rho, Q)$*-indistinguishability,* $(\frac{\varepsilon}{4n}, \rho, Q)$*-index hiding, and* ρ*-message privacy, and simulability of partial decryptions for functions* $\rho = \rho(\lambda, n, t)$*,* $\varepsilon = 1/\mathsf{poly}(\lambda)$*, and* $Q = 2^7 3n^2\lambda/\varepsilon^2$*. Then, instantiating Construction 1 with* Π *yields a* TT-KEM *that is traceable against* ρ *adaptive partial insiders with limit* ε*.*

Specifically, if Π satisfies the above conditions for $\rho = t - 1$ (as guaranteed by our TPLBE-PD construction in Section 6), then Π' is *maximally* resilient against adaptive partial insiders.

Traceability Proof Sketch. We first deal with the partial decryption oracle for valid ciphertexts by replacing the real partial decryptions returned by $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ with simulated ones through SimPD. By the simulability of partial decryptions of the underlying TPLBE-PD scheme, these are computationally indistinguishable, and thus cannot significantly change the probabilities of the events GoodTr, BadTr, and GoodDec.

Unframability. We prove unframability (i.e., $\Pr[\mathsf{BadTr}] \leq \mathsf{negl}(\lambda)$) by a reduction to index hiding. To break unframability, there must exist a party i^* that is (a) not corrupted by the adversary, and (b) tracing the decoder D causes $\hat{p}_{i^*-1} - \hat{p}_{i^*} \geq \varepsilon/4n$. The probability of this happening can be bound by a simple hybrid argument across all n parties that performs a reduction to the $(\frac{\varepsilon}{4n}, \rho, Q)$-index-hiding game of the underlying TPLBE-PD scheme.

Tracing good Decoders accuses at least one real traitor. The remainder of our proof proceeds in 2 steps.

First, we show that $\hat{p}_n \geq \frac{1}{2} + \frac{\varepsilon}{8}$ can only happen with negligible probability. To see this, recall that $\hat{p}_n$ is the decoder's measured success rate across Q repetitions of distinguishing ciphertexts where all parties are revoked. Since Q is sufficiently large, a simple Chernoff bound implies that the decoder's real success probability p_n cannot be significantly smaller than $\hat{p}_n$, except with negligible probability. Such a decoder can then be used by a reduction to win the message privacy game of TPLBE-PD with related advantage. At this step, it is crucial that our message privacy game allows us to adaptively query for simulated partial decryptions on the challenge ciphertext. This allows the reduction to run the decoder on the challenge ciphertext, supply simulated partial decryptions, and compare the outputs to other responses to guess whether the challenge key is encapsulated within the challenge ciphertext.

Second, we argue that if D is a good decoder, i.e., $P(D) \geq \frac{1}{2} + \varepsilon$, then it must hold that $\hat{p}_0 \geq \frac{1}{2} + \frac{3\varepsilon}{8}$ except with negligible probability. Note however, that $P(D)$ and $\hat{p}_0$ are quite different in nature: $P(D)$ is the decoder's success probability for *honestly generated ciphertexts with real partial decryptions* (i.e., ciphertexts stemming from Enc and partial decryptions stemming from PDec), while $\hat{p}_0$ is an empirical estimate of the decoder's success probability for $\mathsf{TrEnc}(0)$ with partial decryptions stemming from SimPD. To close this gap, we first define $P_{\mathsf{Sim}}(D)$ identically to $P(D)$, except that the decoder now receives simulated instead of real partial decryptions. A simple reduction to simulability of partial decryptions of TPLBE-PD allows us to bound $P(D) - P_{\mathsf{Sim}}(D) \leq \varepsilon/4$ except with negligible probability. If $P_{\mathsf{Sim}}(D) \geq \frac{1}{2} + \frac{3\varepsilon}{4}$, then, by a Chernoff bound, testing the decoder Q times with Enc ciphertexts and simulated partial decryptions, results in a success rate of $\hat{q} \geq \frac{1}{2} + \frac{\varepsilon}{2}$, except with negligible probability. If an adversary can produce decoders for which $\hat{q} \geq \frac{1}{2} + \frac{\varepsilon}{2}$, but $\hat{p}_0 < \frac{1}{2} + \frac{3\varepsilon}{8}$ (i.e., $\hat{q} - \hat{p}_0 > \frac{\varepsilon}{8}$), then this adversary can be used by a straight-forward reduction to break $(\frac{\varepsilon}{8}, \rho, Q)$-indistinguishability of TPLBE-PD.

Putting together the above arguments shows that tracing good decoders results in $\hat{p}_0 \geq \frac{1}{2} + \frac{3\varepsilon}{8}$ but $\hat{p}_n < \frac{1}{2} + \frac{\varepsilon}{8}$ except with negligible probability. Then, by pigeon-hole principle, there must exist at least one $i^* \in [n]$ such that $\hat{p}_{i^*-1} - \hat{p}_{i^*} \geq \frac{\varepsilon}{4n}$, which is consequently accused as a traitor. Note that due to unframability, i^* must be a real traitor except with negligible probability.

We next formalize the above.

5.1 A Formal Proof of Theorem 5.1

Proof of Theorem 5.1. Fix an arbitrary $\varepsilon = 1/\mathsf{poly}(\lambda)$ and ρ such that $0 \leq \rho < t$. Let Π' be Construction 1 instantiated with a TPLBE-PD scheme Π as in Theorem 5.1. Consider the traceability game $\mathsf{Game\text{-}Tr}^{\Pi'}(1^\lambda, \varepsilon, \rho)$ instantiated with Π', and let $\mathcal{A}$ be an arbitrary PPT adversary against this game. W.l.o.g., assume that $\mathcal{A}$ initially outputs $n, t \in \mathsf{poly}(\lambda)$ such that $0 < t \leq n$.

Replacing $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ with simulated partial decryptions. First, we make a modification to the tracing game Game-Tr, where we replace the oracle $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ with an oracle $\mathcal{O}^{\mathsf{SimPD}}$ that, given (i, r_{Enc}), computes $(k, c) \leftarrow \Pi.\mathsf{Enc}(\mathsf{pk}; r_{\mathsf{Enc}})$, and outputs simulated partial decryptions as $d_i \leftarrow \Pi.\mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$ instead of real partial decryptions. Due to simulability of partial decryptions of Π (Definition 4.2), the outcome of the experiment should change at most by a negligible amount. More formally, consider the following Lemma.

Lemma 5.2 *Let* $\mathsf{GoodTr}'_{\mathcal{A}}$, $\mathsf{GoodDec}'_{\mathcal{A}}$, *and* $\mathsf{BadTr}'_{\mathcal{A}}$ *denote the respective events of the traceability game, when run with the* $\mathcal{O}^{\mathsf{SimPD}}$ *oracle instead of the* $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ *oracle. There exists a reduction* $\mathcal{B}$ *such that all of the following hold*

$$\begin{aligned}\left|\Pr[\mathsf{GoodTr}_{\mathcal{A}}] - \Pr\left[\mathsf{GoodTr}'_{\mathcal{A}}\right]\right| &\leq 2 \cdot \mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda),\\ \left|\Pr[\mathsf{GoodDec}_{\mathcal{A}}] - \Pr\left[\mathsf{GoodDec}'_{\mathcal{A}}\right]\right| &\leq 2 \cdot \mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda), \text{ and}\\ \left|\Pr[\mathsf{BadTr}_{\mathcal{A}}] - \Pr\left[\mathsf{BadTr}'_{\mathcal{A}}\right]\right| &\leq 2 \cdot \mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda).\end{aligned}$$

Proof of Lemma 5.2. First, observe that, for $E \in \{\mathsf{GoodTr}, \mathsf{GoodDec}, \mathsf{BadTr}\}$, the events E and E' can only differ, if $\mathcal{A}$ queries the oracle with some $(i, r_{\mathsf{Enc}}) \in [n] \times \mathcal{R}$, such that $\mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c) \neq \mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$, where $(k, c) \leftarrow \mathsf{Enc}(pk; r_{\mathsf{Enc}})$. Denote with Hit the event that $\mathcal{A}$ queries the oracle with such an input.

We will use $\mathcal{A}$ to construct a PPT reduction $\mathcal{B}$ that wins $\mathsf{GameSimPD}$ with advantage $\Pr[\mathsf{Hit}_{\mathcal{A}}]$.

The reduction $\mathcal{B}$, running in $\mathsf{GameSimPD}$, simulates $\mathsf{Game\text{-}Tr}$ to $\mathcal{A}$ as follows: Initially, $\mathcal{B}$ controls all parties $\{\mathbb{P}_i\}_{i\in[n]}$ during the Setup protocol, and simulates their behavior honestly. Whenever $\mathcal{A}$ chooses to corrupt a party through the $\mathcal{O}^{\mathsf{corrupt}}$ oracle, $\mathcal{B}$ yields control over that party to $\mathcal{A}$, forwarding its secret state up to that point. $\mathcal{B}$ then forwards all messages to and from corrupted parties between the challenger and $\mathcal{A}$ until the end of the Setup protocol. For every round of the Setup protocol that is executed by a party corrupted by $\mathcal{A}$, $\mathcal{B}$ receives from $\mathcal{A}$ the randomness to execute that round, and simulates execution for $\mathcal{A}$ (see Remark 6.1). As a result, at the end of Setup, $\mathcal{B}$ knows all secret keys $\{\mathsf{sk}_i\}_{i\in[n]}$.

Simulating $\mathcal{O}^{\mathsf{PDec\text{-}Val}}/\mathcal{O}^{\mathsf{SimPD}}$. On input (i, r_{Enc}), $\mathcal{B}$ computes $(k, c) \leftarrow \Pi.\mathsf{Enc}(\mathsf{pk}, r_{\mathsf{Enc}})$. Then $\mathcal{B}$ computes the partial decryptions $d_i \leftarrow \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)$, as well as the simulated partial decryptions $d'_i \leftarrow \mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$. If $d_i = d'_i$, then $\mathcal{B}$ just outputs d_i. If, however $d_i \neq d'_i$ (i.e., the event Hit happens), then $\mathcal{B}$ aborts the simulation and queries the Chal oracle of its challenger on input r_{Enc}, receiving $\{d^*_j\}_{j\in[n]}$. If $d^*_i = d_i$, $\mathcal{B}$ outputs $b' = 0$ guessing that the challenger sent real partial decryptions. If, however $d^*_i = d'_i$, then $\mathcal{B}$ outputs $b' = 1$, guessing that the challenger sent simulated partial decryptions.

If the adversary outputs a decoder D without ever triggering the eveng Hit, then $\mathcal{B}$ just outputs a random bit to its challenger.

Analysis. Observe that if Hit happens, then $\mathcal{B}$ always wins $\mathsf{GameSimPD}$. If, however Hit never happens, then $\mathcal{B}$ wins $\mathsf{GameSimPD}$ with probability $1/2$. Thus, we calculate $\mathcal{B}$'s advantage as follows:

$$\mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda) = \left|\Pr[\mathsf{Hit}_{\mathcal{A}}] \cdot 1 + (1 - \Pr[\mathsf{Hit}_{\mathcal{A}}]) \cdot \frac{1}{2} - \frac{1}{2}\right| = \frac{\Pr[\mathsf{Hit}_{\mathcal{A}}]}{2}.$$

We get the following bound for all $E \in \{\mathsf{GoodTr}, \mathsf{GoodDec}, \mathsf{BadTr}\}$:

$$|\Pr[E_{\mathcal{A}}] - \Pr[E'_{\mathcal{A}}]| \leq \Pr[\mathsf{Hit}_{\mathcal{A}}] \leq 2 \cdot \mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda),$$

which concludes the proof. ∎

In the remainder of the proof, we will consider the traceability game with respect to on $\mathcal{O}^{\mathsf{SimPD}}$ oracle instead of the $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ oracle unless explicitly stated otherwise.

Unframability. We first consider unframability. Intuitively, $\mathcal{A}$ can only frame an honest party i, by not corrupting i ($i \notin \mathcal{T}'$), and outputting a decoder D such that $\hat{p}_{i-1} - \hat{p}_i \geq \varepsilon/4n$ during tracing. This, however, contradicts the index-hiding property of Π.

Lemma 5.3 *There exists a PPT reduction* $\mathcal{B}$ *such that*

$$\Pr[\mathsf{BadTr}'_{\mathcal{A}}] \leq n \cdot \mathsf{Adv}^{\mathsf{IHEmp}}_{\mathcal{B},\Pi}(1^\lambda, \varepsilon/4n, \rho, Q).$$

Proof of Lemma 5.3. Consider the following PPT reduction $\mathcal{B}$ that internally uses $\mathcal{A}$ to win the IHEmp experiment of Π'.

After receiving n, t from $\mathcal{A}$, $\mathcal{B}$ initially picks a random $i^* \xleftarrow{\$} [n]$, and forwards (n, t, i^*) to its IHEmp challenger, which means that $\mathcal{B}$ controls the parties $[n] \setminus \{i^*\}$ in the IHEmp experiment, while simulating $\mathsf{Game\text{-}Tr}$ to $\mathcal{A}$. Henceforth, we say that the guess i^* is "correct", whenever $\mathcal{A}$ causes $\hat{p}_{i^*-1} - \hat{p}^*_i \geq \varepsilon/4n$ without corrupting i^*.

Simulating Setup. Setup is a (possibly interactive) protocol between a trusted party $\mathbb{D}$, a tracer $\mathbb{T}$, and the parties $\{\mathbb{P}_i\}_{i\in[n]}$. In $\mathcal{B}$'s simulation, the IHEmp challenger takes control of the trusted party $\mathbb{D}$ as well as the tracer $\mathbb{T}$. The challenger also controls party i^*, while $\mathcal{B}$ controls all parties $[n] \setminus \{i^*\}$. Whenever $\mathcal{A}$ corrupts a party through the corruption oracle $\mathcal{O}^{\mathsf{corrupt}}$ during the Setup phase, $\mathcal{B}$ yields control and

forwards the secret state of that party to $\mathcal{A}$. Otherwise, $\mathcal{B}$ simulates Setup honestly on behalf of all parties controlled by it, and appropriately forwards messages between its challenger and $\mathcal{A}$.

Simulating $\mathcal{O}^{\mathsf{corrupt}}$. On input i, if $i \neq i^*$, $\mathcal{B}$ forwards the secret state of party i (including sk_i) to $\mathcal{A}$. If, however, $i = i^*$, $\mathcal{B}$ aborts, outputting a dummy decoder to its challenger. Looking ahead, we can see that, if i^* is a correct guess, then $\mathcal{B}$ will never abort.

Simulating $\mathcal{O}^{\mathsf{SimPD}}$. On input (i, r_{Enc}) the reduction $\mathcal{B}$ computes $(k, c) \leftarrow \Pi.\mathsf{Enc}(\mathsf{pk}; r_{\mathsf{Enc}})$ and outputs $d_i \leftarrow \Pi.\mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$ to $\mathcal{A}$.

Once $\mathcal{A}$ outputs a decoder D, $\mathcal{B}$ simply forwards that decoder to its challenger.

Analysis. Whenever $\mathsf{BadTr}'_{\mathcal{A}} = 1$, then there must exist at least one $i \in [n]$ such that $i \notin \mathcal{T}$ (i.e., $\mathcal{A}$ did not corrupt i through $\mathcal{O}^{\mathsf{corrupt}}$), and $\hat{p}_{i-1} - \hat{p}_i \geq \varepsilon/4n$, i.e., there must exist some $i \in [n]$ that is a correct guess, and thus, if $\mathsf{BadTr}'_{\mathcal{A}} = 1$, $\mathcal{B}$ guesses i^* correctly with probability at least $1/n$. Also, observe that whenever $i \notin \mathcal{T}$, $\mathcal{B}$ perfectly simulates $\mathsf{Game\text{-}Tr}$ to $\mathcal{A}$.

Now, consider the quantities a and b computes by $\mathcal{B}$'s IHEmp challenger, and observe that a is computed identically to $\hat{p}_{i^*-1}$, and b is computed identically to $\hat{p}_{i^*}$. Thus, whenever $i^* \notin \mathcal{T}$, it holds that $|a - b| \geq \delta$ for $\delta = \varepsilon/4n$ is equivalent to $\hat{p}_{i^*-1} - \hat{p}_{i^*} \geq \varepsilon/4n$, in which case $\mathcal{B}$ wins the IHEmp experiment, and we get that

$$\Pr[\mathsf{BadTr}'_{\mathcal{A}}] \leq n \cdot \mathsf{Adv}^{\mathsf{IHEmp}}_{\mathcal{B},\Pi}(1^\lambda, \varepsilon/4n, \rho, Q).$$

■

Traceability. The remainder of the proof will show that $\Pr\big[\mathsf{GoodTr}'_{\mathcal{A}}\big] \geq \Pr\big[\mathsf{GoodDec}'_{\mathcal{A}}\big] - \mathsf{negl}(\lambda)$. We first introduce the following lemma.

Lemma 5.4 *There exists a PPT reduction $\mathcal{B}$ and a negligible function negl such that*

$$\Pr\left[\hat{p}_n \geq \frac{1}{2} + \frac{\varepsilon}{8}\right] \leq \frac{128}{\varepsilon^2} \cdot \mathsf{Adv}^{\mathsf{MP}}_{\mathcal{B},\Pi}(\lambda, \rho) + \mathsf{negl}(\lambda).$$

Proof. First, denote with p_n the random variable capturing the decoder D's success probability of distinguishing for $\mathsf{TrEnc}(\mathsf{tk}, n)$ when given simulated partial decryptions. More formally, let

$$p_n \coloneqq \Pr\left[D(\mathsf{st}, \{d_i\}_{i \in \mathcal{I}_{\mathsf{pd}}}) = b \colon \begin{array}{c} ((k_0, c); r_{\mathsf{Enc}}) \xleftarrow{\$} \Pi.\mathsf{TrEnc}(\mathsf{tk}, n) \\ k_1 \xleftarrow{\$} \mathcal{K}; b \xleftarrow{\$} \{0,1\} \\ (\mathcal{I}_{\mathsf{pd}}, \mathsf{st}) \xleftarrow{\$} D(k_b, c) \\ \{d_i\}_{i \in [n]} \leftarrow \mathsf{SimPDs}(\mathcal{I}_{\mathsf{pd}}, \rho, \mathsf{pk}, c, r_{\mathsf{Enc}}) \end{array} \right].$$

Since $\hat{p}_n$ computes the fraction of successful runs of the decoder D in the experiment captured by p_n across Q repetitions, it must hold that $\hat{p}_n$ and p_n cannot fall too far apart, except with negligible probability.

Claim 5.5 It holds that

$$\Pr\left[\hat{p}_n \geq \frac{1}{2} + \frac{\varepsilon}{8}\right] \leq \Pr\left[p_n \geq \frac{1}{2} + \frac{\varepsilon}{16}\right] + \exp(-\lambda).$$

Proof of Claim 5.5. First, observe that, by the law of total probabilities, it holds that

$$\begin{aligned} \Pr\left[\hat{p}_n \geq \frac{1}{2} + \frac{\varepsilon}{8}\right] &\leq \Pr\left[\hat{p}_n \geq \frac{1}{2} + \frac{\varepsilon}{8} \wedge p_n \geq \frac{1}{2} + \frac{\varepsilon}{16}\right] + \Pr\left[\hat{p}_n \geq \frac{1}{2} + \frac{\varepsilon}{8} \wedge p_n < \frac{1}{2} + \frac{\varepsilon}{16}\right] \\ &\leq \Pr\left[p_n \geq \frac{1}{2} + \frac{\varepsilon}{16}\right] + \Pr\left[\hat{p}_n \geq \frac{1}{2} + \frac{\varepsilon}{8} \middle| p_n < \frac{1}{2} + \frac{\varepsilon}{16}\right]. \end{aligned}$$

Hence, to prove the claim, it remains to show that if $p_n < \frac{1}{2} + \frac{\varepsilon}{16}$, then $\hat{p}_n \geq \frac{1}{2} + \frac{\varepsilon}{8}$ can happen at most with probability $\exp(-\lambda)$. Let X_i denote the outcome of a single repetition, and thus $\hat{p}_n = \sum_{i=1}^{Q} X_i$, and let $\mu = Q \cdot (\frac{1}{2} + \frac{\varepsilon}{16})$, then a simple Hoeffding bound shows that

$$\begin{aligned} \Pr\left[\hat{p}_n \geq \frac{1}{2} + \frac{\varepsilon}{8} \middle| p_n < \frac{1}{2} + \frac{\varepsilon}{16}\right] &\leq \Pr\left[\hat{p}_n \geq \mu + \frac{Q\varepsilon^2}{16}\right] \\ &\leq \exp\left(-\frac{Q\varepsilon}{8}\right) \overset{Q \geq 8\lambda/\varepsilon^2}{\leq} \exp(-\lambda). \end{aligned}$$

■

Next, we construct a PPT reduction $\mathcal{B}$ running in the MP experiment such that

$$\Pr\left[p_n \geq \frac{1}{2} + \frac{\varepsilon}{16}\right] \leq \frac{128}{\varepsilon^2} \cdot \mathsf{Adv}^{\mathsf{MP}}_{\mathcal{B},\Pi}(\lambda, \rho).$$

As a shorthand, denote $\gamma = \Pr\left[p_n \geq \frac{1}{2} + \frac{\varepsilon}{16}\right]$.

The reduction $\mathcal{B}$ simulates $\mathsf{Game\text{-}Tr}$ perfectly to $\mathcal{A}$ up to the point where $\mathcal{A}$ outputs a decoder D. Note that $\mathcal{B}$ can do so, because in the MP experiment, it controls *all* parties, and can thus simulate arbitrary $\mathcal{O}^{\mathsf{corrupt}}$ queries to $\mathcal{A}$. Similarly, $\mathcal{B}$ can simulate $\mathcal{O}^{\mathsf{SimPD}}(i, r_{\mathsf{Enc}})$ queries trivially. At the end of the Setup phase, $\mathcal{B}$ receives a tuple (c^*, k^*) from its challenger.

Next, $\mathcal{B}$ samples $r_{\mathsf{Enc}} \xleftarrow{\$} \mathcal{R}$, and queries its $\mathsf{TrEnc}(\mathsf{tk}, \cdot; \cdot)$ oracle on input $(n; r_{\mathsf{Enc}})$, receiving (k_0, c) as a response. $\mathcal{B}$ then samples $k_1 \xleftarrow{\$} \mathcal{K}$ und queries the decoder as follows:

- Sample $\beta \xleftarrow{\$} \{0, 1\}$
- Query $(\mathcal{I}_{\mathsf{pd}}, \mathsf{st}) \xleftarrow{\$} D(k_\beta, c)$
- Compute $d_i \leftarrow \mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$ for all $i \in \mathcal{I}_{\mathsf{pd}}$, setting all d_i to $\perp$ if $\mathcal{I}_{\mathsf{pd}} \not\subseteq [n]$ or $|\mathcal{I}_{\mathsf{pd}}| > \rho$.
- Query $o_{\mathsf{ref}} \xleftarrow{\$} D(\mathsf{st}, \{d_i\}_{i \in \mathcal{I}_{\mathsf{pd}}})$

Then, run $(\mathcal{I}_{\mathsf{pd}}{}^*, \mathsf{st}^*) \xleftarrow{\$} D(k^*, c^*)$, and forward $\mathcal{I}_{\mathsf{pd}}{}^*$ to the MP experiment. As a response, $\mathcal{B}$ receives $\{d_i^*\}_{i \in \mathcal{I}_{\mathsf{pd}}{}^*}$, which are the simulated partial decryptions on behalf of $\mathcal{I}_{\mathsf{pd}}{}^*$ on the challenge ciphertext c^*. Then, $\mathcal{B}$ queries $o_{\mathsf{chal}} \xleftarrow{\$} D(\mathsf{st}^*, \{d_i^*\}_{i \in \mathcal{I}_{\mathsf{pd}}{}^*})$, and outputs

$$b' = \begin{cases} \beta \text{ if } o_{\mathsf{chal}} = o_{\mathsf{ref}} \\ 1 - \beta \text{ else} \end{cases}$$

to its challenger.

Analysis. Denote with b the internal bit of the MP challenger, and observe that $\mathcal{B}$ correctly guesses $b' = b$ exactly if the decoder either guesses the bit of both the reference and the challenge ciphertext correctly, or if the decoder guesses the bit of both the reference and the challenge ciphertext incorrectly. Given that $\mathcal{B}$ guesses correctly for any $\mathsf{TrEnc}(\mathsf{tk}, n)$ ciphertext with probability p_n, we get that

$$\Pr[b' = b] = p_n^2 + (1 - p_n)^2$$

Observe that, for all $0 \leq p_n \leq 1$ it holds that $p_n^2 + (1 - p_n)^2 \geq 1/2$. Given that $\mathcal{A}$ outputs a decoder that achieves $p_n \geq \frac{1}{2} + \frac{\varepsilon}{16}$ with probability γ, we get

$$\begin{aligned} \mathsf{Adv}^{\mathsf{MP}}_{\mathcal{B},\Pi}(\lambda, \rho) &\geq \left(\gamma \cdot \Pr\left[b' = b \middle| p_n \geq \frac{1}{2} + \frac{\varepsilon}{16}\right] + (1 - \gamma) \cdot \frac{1}{2}\right) - \frac{1}{2} \\ &\geq \gamma \left(\left(\frac{1}{2} + \frac{\varepsilon}{16}\right)^2 + \left(\frac{1}{2} - \frac{\varepsilon}{16}\right)^2\right) - \frac{\gamma}{2} \\ &= \gamma \cdot \frac{\varepsilon^2}{128} \end{aligned}$$

Rearranging and applying Claim 5.5, we get that

$$\Pr\left[\hat{p}_n \geq \frac{1}{2} + \frac{\varepsilon}{8}\right] \leq \gamma + \exp(-\lambda) \leq \frac{128}{\varepsilon^2} \cdot \mathsf{Adv}^{\mathsf{MP}}_{\mathcal{B},\Pi}(\lambda, \rho) + \exp(-\lambda),$$

which concludes the proof of Lemma 5.4. ∎

Our next goal is to show that, whenever $P(D) \geq 1/2 + \varepsilon$, then $\hat{p}_0 \geq 1/2 + \frac{3\varepsilon}{8}$ except with negligible probability. Note however, that $P(D)$ and $\hat{p}_0$ are quite different in nature: $P(D)$ is the decoder's success probability for *honestly generated ciphertexts with real partial decryptions* (i.e., ciphertexts stemming from Enc and partial decryptions stemming from PDec), while $\hat{p}_0$ is an empirical estimate of the decoder's success probability for $\mathsf{TrEnc}(0)$ with partial decryptions stemming from SimPD, i.e., simulated partial decryptions. We close this gap in two steps: Denote with $P_{\mathsf{Sim}}(D)$ the decoder's success probability against Enc ciphertexts with simulated partial decryptions. First, we show that the probability that $P(D) - P_{\mathsf{Sim}}(D) > \varepsilon/4$ is negligible, by simulability of partial decryptions of the $\mathsf{TPLBE\text{-}PD}$ scheme Π.

Then, we argue that, by indistinguishability of Π, if $P_{\mathsf{Sim}}(D) \geq 1/2 + \frac{3\varepsilon}{4}$, then $\hat{p}_0$ must be at least $1/2 + \frac{3\varepsilon}{8}$, except with negligible probability. Intuitively, this is because it should be hard for adversaries to produce a decoder that significantly distinguishes between Enc ciphertexts with simulated partial decryptions and TrEnc ciphertexts with simulated partial decryptions.

We start with the first step, applying simulability of partial decryptions of Π (Definition 4.2).

Lemma 5.6 *There exists a PPT reduction $\mathcal{B}$ such that*

$$\Pr\left[|P(D) - P_{\mathsf{Sim}}(D)| > \frac{\varepsilon}{4}\right] \leq \frac{8}{\varepsilon}\mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda).$$

Proof of Lemma 5.6. The reduction $\mathcal{B}$, while running in $\mathsf{GameSimPD}$ (Fig. 4), simulates $\mathsf{Game\text{-}Tr}$ identically to the reduction to message privacy above up to the point where $\mathcal{A}$ outputs a decoder D. $\mathcal{B}$ then samples $r_{\mathsf{Enc}} \xleftarrow{\$} \mathcal{R}$, computes $(k_0, c) \xleftarrow{\$} \Pi.\mathsf{Enc}(\mathsf{pk}; r_{\mathsf{Enc}})$, and queries the challenge oracle Chal with r_{Enc}. As a result, the reduction receives partial decryptions $\{d_i\}_{i\in[n]}$. If the internal bit b of the $\mathsf{GameSimPD}$ challenger is 0, then the partial decryptions stem from PDec, and otherwise the partial decryptions stem from SimPD.

Then, the reduction samples a random bit β as well as $k_1 \xleftarrow{\$} \mathcal{K}$, and queries $D(k_\beta, c)$, receiving $(\mathsf{st}, \mathcal{I}_{\mathsf{pd}})$ as a response. If $\mathcal{I}_{\mathsf{pd}} \not\subseteq [n]$ or $|\mathcal{I}_{\mathsf{pd}}| > \rho$, $\mathcal{B}$ overwrites $d_i \leftarrow \perp$ for all $i \in \mathcal{I}_{\mathsf{pd}}$.

Finally, $\mathcal{B}$ outputs $b' = \beta \oplus D(\mathsf{st}, \{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}})$.

Analysis. Observe that, if $b = 0$, then $\mathcal{B}$ outputs $b' = 0$ with probability $P(D)$. If $b' = 1$, then $\mathcal{B}$ outputs $b' = 1$ with probability $1 - P_{\mathsf{Sim}}(D)$, and thus

$$\Pr[b' = b] = \frac{1}{2} \cdot P(D) + \frac{1}{2}(1 - P_{\mathsf{Sim}}(D)) = \frac{1}{2} + \frac{P(D) - P_{\mathsf{Sim}}(D)}{2}.$$

Denote with γ the probability that $|P(D) - P_{\mathsf{Sim}}(D)| \geq \varepsilon/4$ for the decoder outputs by $\mathcal{A}$.

We get that

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda) &\geq \gamma \cdot \Pr\left[b' = b \middle| |P(D) - P_{\mathsf{Sim}}(D)| > \frac{\varepsilon}{4}\right] \\
&\quad + (1-\gamma)\Pr\left[b' = b \middle| |P(D) - P_{\mathsf{Sim}}(D)| \leq \frac{\varepsilon}{4}\right] - \frac{1}{2} \\
&\geq \gamma \cdot \left(\frac{1}{2} + \frac{\varepsilon}{8}\right) + (1-\gamma) \cdot \frac{1}{2} - \frac{1}{2} \\
&\geq \gamma\frac{\varepsilon}{8}.
\end{aligned}$$

Rearranging, we can bound

$$\Pr\left[|P(D) - P_{\mathsf{Sim}}(D)| > \frac{\varepsilon}{4}\right] \leq \frac{8}{\varepsilon}\mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda).$$

■

Next, consider the following hypothetical experiment given a decoder D. Let $\hat{q}$ count the fraction of how many times the decoder D correctly guesses the bit b when queried with an Enc ciphertext and partial decryptions from SimPD across Q repetitions. Observe that this experiment is essentially an empirical estimate of $P_{\mathsf{Sim}}(D)$, and thus should not fall significantly short of $P_{\mathsf{Sim}}(D)$.

Claim 5.7 If $P_{\mathsf{Sim}}(D) \geq 1/2 + \frac{3\varepsilon}{4}$, then $\Pr\left[\hat{q} < \frac{1}{2} + \frac{\varepsilon}{2}\right] \leq \exp(-\lambda)$.

Proof of Claim 5.7. The claim follows directly from Hoeffding's inequality.

$$\Pr\left[\hat{q} < \frac{1}{2} + \frac{\varepsilon}{2} \middle| P_{\mathsf{Sim}}(D) \geq 1/2 + \frac{3\varepsilon}{4}\right] \leq \exp\left(-\frac{Q\varepsilon^2}{8}\right) \overset{Q\geq 8\lambda/\varepsilon^2}{\leq} \exp(-\lambda),$$

which is negligible for our choice of Q. ■

Next, we can apply indistinguishability of Π (Definition 4.3) to argue that $|\hat{q} - \hat{p}_0| < \varepsilon/8$ except with negligible probability.

Lemma 5.8 *There exists a PPT reduction $\mathcal{B}$ such that*

$$\Pr\left[|\hat{q} - \hat{p}_0| \geq \frac{\varepsilon}{8}\right] \leq \mathsf{Adv}^{\mathsf{IndEmp}}_{\mathcal{B},\Pi}(\lambda, \frac{\varepsilon}{8}, \rho, Q).$$

Proof of Lemma 5.8. This proof is very similar to our reduction to index hiding (see proof of Lemma 5.3). In particular, the reduction $\mathcal{B}$ simulates $\mathsf{Game\text{-}Tr}$ to the adversary $\mathcal{A}$ up to the point where $\mathcal{A}$ outputs a decoder D.

Simulating Setup. $\mathcal{B}$ simulates Setup by forwarding messages between its challenger and $\mathcal{A}$ for all parties corrupted by $\mathcal{A}$ through $\mathcal{O}^{\mathsf{corrupt}}$.

Simulating $\mathcal{O}^{\mathsf{corrupt}}$. Whenever $\mathcal{A}$ issues a corruption query for party i, $\mathcal{B}$ forwards the secret state of that party (including sk_i) to $\mathcal{A}$. Should the corruption query happen during the execution of the Setup protocol, $\mathcal{B}$ additionally yields control over that party to $\mathcal{A}$, and forwards messages to and from that party between the $\mathcal{A}$ and the challenger.

Simulating $\mathcal{O}^{\mathsf{SimPD}}$. $\mathcal{B}$ simulates $\mathcal{O}^{\mathsf{SimPD}}$ perfectly according to protocol. On input (i, r_{Enc}), $\mathcal{B}$ computes $(k, c) \leftarrow \Pi.\mathsf{Enc}(\mathsf{pk}; r_{\mathsf{Enc}})$, and then sends $d_i \leftarrow \mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$ to $\mathcal{A}$. Once $\mathcal{A}$ outputs a decoder D, $\mathcal{B}$ forwards D to its challenger.

Analysis. Consider the quantities a and b computed by $\mathcal{B}$'s IndEmp challenger, and observe that a is computed identically to $\hat{q}$ and b is computed identically to $\hat{p}_0$. Thus, it holds that $|\hat{q} - \hat{p}_0| \geq \frac{\varepsilon}{8}$ is equivalent to $|a - b| \geq \delta$ for $\delta = \varepsilon/8$, and we get that

$$\Pr\left[|\hat{q} - \hat{p}_0| \geq \frac{\varepsilon}{8}\right] = \mathsf{Adv}^{\mathsf{IndEmp}}_{\mathcal{B},\Pi}(\lambda, \frac{\varepsilon}{8}, \rho, Q),$$

which concludes the proof. ∎

Putting it all together. Together, Lemma 5.6, Claim 5.7, Lemma 5.8, and Lemma 5.4 show that the event $\mathsf{GoodDec}'$ implies that $\hat{p}_0 \geq \frac{1}{2} + \frac{\varepsilon}{2}$ and $\hat{p}_n < \frac{1}{2} + \frac{\varepsilon}{8}$ except with negligible probability. More formally, it holds that

$$\Pr\left[\mathsf{GoodDec}'_{\mathcal{A}}\right] \leq \Pr\left[\hat{p}_0 \geq \frac{1}{2} + \frac{\varepsilon}{2} \wedge \hat{p}_n < \frac{1}{2} + \frac{\varepsilon}{8}\right] + \delta,$$

where $\delta = \frac{8}{\varepsilon}\mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda) + \mathsf{Adv}^{\mathsf{IndEmp}}_{\mathcal{B},\Pi}(\lambda, \frac{\varepsilon}{8}, \rho, Q)\frac{128}{\varepsilon^2} \cdot \mathsf{Adv}^{\mathsf{MP}}_{\mathcal{B},\Pi}(\lambda, \rho) + 2\exp(-\lambda)$.

Now, if $\hat{p}_0 \geq \frac{1}{2} + \frac{\varepsilon}{2}$, but $\hat{p}_n < \frac{1}{2} + \frac{\varepsilon}{8}$, then it holds that $\hat{p}_0 - \hat{p_n} > \frac{\varepsilon}{4}$. By a pigeonhole principle, there must exist at least one $i \in [n]$ such that $\hat{p}_{i-1} - \hat{p}_i \geq \frac{\varepsilon}{4n}$, in which case the tracing algorithm will have $i \in \mathcal{T}'$, and thus $\mathcal{T}' \neq \emptyset$. Additionally, observe that it must hold that $\mathcal{T}' \subseteq \mathcal{T}$ except when BadTr' happens. Taking into account our assumptions in Theorem 5.1, we get that

$$\begin{aligned}
\Pr\left[\mathsf{GoodDec}'_{\mathcal{A}}\right] &\leq \Pr\left[\hat{p}_0 - \hat{p_n} > \frac{\varepsilon}{4}\right] + \delta \\
&\leq \Pr\left[\mathsf{GoodTr}'_{\mathcal{A}}\right] + \Pr[\mathsf{BadTr}'_{\mathcal{A}}] - \delta \\
&\leq \Pr\left[\mathsf{GoodTr}'_{\mathcal{A}}\right] + n \cdot \mathsf{Adv}^{\mathsf{IHEmp}}_{\mathcal{B},\Pi}(1^\lambda, \varepsilon/4n, \rho, Q) + \delta.
\end{aligned}$$

Finally, we can apply Lemma 5.2 and rearrange to get the following for appropriate reductions $\mathcal{B}$:

$$\begin{aligned}
\Pr[\mathsf{GoodTr}_{\mathcal{A}}] &\geq \Pr\left[\mathsf{GoodTr}'_{\mathcal{A}}\right] - 2 \cdot \mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda) \\
&\geq \Pr\left[\mathsf{GoodDec}'_{\mathcal{A}}\right] - n \cdot \mathsf{Adv}^{\mathsf{IHEmp}}_{\mathcal{B},\Pi}(1^\lambda, \varepsilon/4n, \rho, Q) - \delta - 2 \cdot \mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda) \\
&\geq \Pr[\mathsf{GoodDec}_{\mathcal{A}}] - n \cdot \mathsf{Adv}^{\mathsf{IHEmp}}_{\mathcal{B},\Pi}(1^\lambda, \varepsilon/4n, \rho, Q) - \delta - 4 \cdot \mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda) \\
&\geq \Pr[\mathsf{GoodDec}_{\mathcal{A}}] - \mathsf{negl}(\lambda),
\end{aligned}$$

where in the last step, we applied the assumptions from Theorem 5.1. Similarly, it holds that

$$\begin{aligned}
\Pr[\mathsf{BadTr}_{\mathcal{A}}] &\leq \Pr\left[\mathsf{BadTr}'_{\mathcal{A}}\right] + 2 \cdot \mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda) \\
&\leq n \cdot \mathsf{Adv}^{\mathsf{IHEmp}}_{\mathcal{B},\Pi}(1^\lambda, \varepsilon/4n, \rho, Q) + 2 \cdot \mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{B},\Pi}(\lambda) \leq \mathsf{negl}(\lambda).
\end{aligned}$$

This concludes the proof.

∎

Finally, the traceability transform trivially maintains the strong semantic security of the underlying $\mathsf{TPLBE\text{-}PD}$ scheme.

Lemma 5.9 *If the* $\mathsf{TPLBE\text{-}PD}$ *scheme* Π *is correct and achieves* $\mathsf{SIND\text{-}CPA}$ *security, then instantiating Construction 1 with* Π *yields a* $\mathsf{TT\text{-}KEM}$ *that is correct and* $\mathsf{SIND\text{-}CPA}$ *secure.*

6 Our TPLBE-PD Construction

We next formalize the construction outlined in Section 2.4.

Notation. For a preprocessing key $K = \left([\beta]_1, \mathbf{A}, \mathbf{B}\right)$, we use the following notation to access its components. In particular, we write $K.[\beta]_1 := [\beta]_1, K.A_i^{(q)} := A_i^{(q)}$, and $K.B_i^{(q)} := B_i^{(q)}$.

Construction 2. The construction is parameterized by $\varepsilon \in (0, 1/2)$ and the number of keys per ciphertext $m \in \mathsf{poly}(\lambda)$. Let Π_{Com} be a commitment scheme.

The $\mathsf{Setup}(1^\lambda, n, t)$ protocol. The protocol relies on a structured CRS, set up in a trusted manner. Given the CRS, the tracer publishes a commitment to the randomness r_{Prep} he will use in the preprocessing phase. Then, every committee member i independently generates $(\mathsf{pk}_i, \mathsf{sk}_i)$, publishing the public key pk_i. Finally, the tracer uses r_{Prep} to generate a preprocessing $\mathcal{U}$ on the committee's public keys $\{\mathsf{pk}_i\}_{i\in[n]}$. Crucially, committee members do not need to interact with each other, and all communication with the tracer occurs through public channels.

$\mathsf{CRS} \xleftarrow{\$} \mathsf{CRSGen}(1^\lambda, n, t)$**.** The scheme uses a structured CRS generated as follows: Initialize a pairing ensemble $E = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \circ, [1]_1, [1]_2, p) \xleftarrow{\$} \mathcal{G}(1^\lambda)$. Setup the commitment scheme $\mathsf{pp}_{\mathsf{Com}} \xleftarrow{\$} \Pi_{\mathsf{Com}}.\mathsf{Setup}(1^\lambda)$. Sample $s \xleftarrow{\$} \mathbb{Z}_p$ and generate a Shamir sharing $\{s_i\}_{i\in[n]} \xleftarrow{\$} \mathsf{Share}(s, n, t)$. Then, Generate n cross-terms by sampling $c_i \xleftarrow{\$} \mathbb{Z}_p^*$ for all $i \in [n]$, and output the CRS as $\mathsf{pp}_{\mathsf{Com}}$, the ensemble E and the group elements:

$$\left([s]_T, \left\{[c_i]_1, [1/c_i]_2, [c_i s_i]_1, \left\{[c_i/c_j]_1\right\}_{j\in[n]\setminus\{i\}}, \sum\nolimits_{j\in[n]\setminus\{i\}} [s_j c_j/c_i]_1\right\}_{i\in[n]}\right)$$

$(\mathsf{com}, r_{\mathsf{Prep}}, \pi^{\mathsf{TrCom}}) \xleftarrow{\$} \mathsf{TrCom}(1^\lambda, n, t, \mathsf{CRS})$**.** Given the CRS, the tracer samples the randomness r_{Prep} he will use to generate a preprocessing $\mathcal{U}$, and commits to it $\mathsf{com} \xleftarrow{\$} \Pi_{\mathsf{Com}}.\mathsf{Com}(r_{\mathsf{Prep}})$, attaching a proof of knowledge π^{TrCom} of the committed randomness r_{Prep} and the opening. The public output is the tuple $(\mathsf{com}, \pi^{\mathsf{TrCom}})$. The private output for the tracer is the randomness r_{Prep}.

$(\mathsf{sk}_i, \mathsf{pk}_i, \pi_i^{\mathsf{KGen}}) \xleftarrow{\$} \mathsf{KGen}(1^\lambda, \mathsf{CRS}, i)$**.** For all $q \in [m]$, party i samples $\alpha_i^{(q)} \xleftarrow{\$} \mathbb{Z}_p^*$, sets $\mathsf{sk}_i \leftarrow \{\alpha_i^{(q)}\}_{q\in[m]}$ and computes from the CRS

$$\mathsf{pk}_i \leftarrow \left(\left\{\left[c_i\alpha_i^{(q)}\right]_1\right\}_{q\in[m]}, \left\{\left[\alpha_i^{(q)} c_i/c_j\right]_1\right\}_{q\in[m], j\in[n]\setminus\{i\}}\right).$$

Furthermore, compute a proof π_i^{KGen} of knowledge of randomness $\{\alpha_i^{(q)}\}_{q\in[m]}$. The private output is sk_i, and the public output is the tuple $(\mathsf{pk}_i, \pi_i^{\mathsf{KGen}})$.

Subroutine $\mathsf{Rand}(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i\in[n]}, \mathcal{R})$

$\beta \xleftarrow{\$} \mathbb{Z}_p^*$

Let $r_i \leftarrow \beta$ for all $i \in [n] \setminus \mathcal{R}$ // Rerandomize with β for all unrevoked parties.

Let $r_i \xleftarrow{\$} \mathbb{Z}_p^*$ for all $i \in \mathcal{R}$ // Rerandomize with new randomness for all revoked parties.

for $i = 1 \ldots n$ **and** $q = 1 \ldots m$ **do**

$A_i^{(q)} \leftarrow r_i \cdot \mathsf{pk}_{i,1}^{(q)}$ // $\equiv \left[c_i \alpha_i^{(q)} r_i\right]_1$

$B_i^{(q)} \leftarrow \sum_{j\in[n]\setminus\{i\}} r_j \cdot \mathsf{pk}_{j,2,i}^{(q)}$ // $\equiv \sum_{j\in[n]\setminus\{i\}} \left[\alpha_j^{(q)} r_j c_j/c_i\right]_1$

return $K \leftarrow \left([\beta]_1, A \coloneqq \{A_i^{(q)}\}_{i\in[n], q\in[m]}, B \coloneqq \{B_i^{(q)}\}_{i\in[n], q\in[m]}\right)$

Figure 8: The rerandomization subroutine Rand.

$(\mathcal{U}, \mathsf{tk}, \pi^{\mathsf{Prep}}) \leftarrow \mathsf{Prep}(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i\in[n]}; r_{\mathsf{Prep}})$**.** The preprocessing algorithm is executed by the tracer using the randomness r_{Prep} committed to in com. Let $N \leftarrow 1024 \cdot (n+1)\lambda m^2 n^2/\varepsilon^2$ be the number of honest and revocation preprocessing keys generated, respectively. Let $d \leftarrow N/(n+1)$. Initialize an empty array $\mathcal{U}_{\mathrm{Norm}}$ of normal preprocessing keys of size N and an array $\mathcal{U}_{\mathrm{TR}}$ of revocation preprocessing keys, of size $(n+1) \times d$. For all $u \in [N]$, compute normal preprocessing keys:

$\mathcal{U}_{\text{Norm}}[u] \xleftarrow{\$} \mathsf{Rand}(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i\in[n]}, \mathcal{R}_0)$. Then, for all $j \in [0,n]$, compute d preprocessing keys each that revoke the set $\mathcal{R}_j$, i.e., for all $\ell \in [d]$, generate: $\mathcal{U}_{\text{TR}}[j,\ell] \xleftarrow{\$} \mathsf{Rand}(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i\in[n]}, \mathcal{R}_j)$. Shuffle the preprocessing keys and assign $\mathcal{U} \xleftarrow{\$} \mathsf{Shuffle}(\mathcal{U}_{\text{Norm}}, \mathcal{U}_{\text{TR}})$. We denote with r_{Prep} the randomness used in the above algorithm, including the randomness in the calls to Rand. Generate a NIZK proof π^{Prep} that $\mathcal{U}$ was generated by Prep on input $(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i\in[n]})$ using the randomness r_{Prep} committed to in TrCom. The public output of Prep is the tuple $(\mathcal{U}, \pi^{\mathsf{Prep}})$, and the private output to the tracer is the tracing key $\mathsf{tk} \leftarrow (\mathcal{U}_{\text{Norm}}, \mathcal{U}_{\text{TR}})$, which stores the mapping between preprocessing keys and revoked sets.

After execution of the above Setup rounds, all proofs π^{TrCom}, $\{\pi_i^{\mathsf{KGen}}\}_{i\in[n]}$, and π^{Prep} are verified. If verification fails, Setup aborts. The output of the setup algorithm are the public parameters $\mathsf{pp} := \mathsf{CRS}$, the public key $\mathsf{pk} := (\mathcal{U}, \{\mathsf{pk}_i\}_{i\in[n]})$, the secret keys $\{\mathsf{sk}_i\}_{i\in[n]}$, and the tracing key tk.

$\mathsf{Enc}(\mathsf{pk})$

$\mathsf{pre} \xleftarrow{\$} \{0,1\}^\lambda$

$((K_1, \mathsf{idx}_1), \ldots, (K_m, \mathsf{idx}_m)) \leftarrow \mathsf{Samp}(\mathcal{U}; \mathcal{H}(\mathsf{pre}))$

return $\mathsf{enc}(\mathsf{CRS}, \{K_q, \mathsf{idx}_q\}_{q\in[m]})$

$\mathsf{TrEnc}(\mathsf{tk}, j)$

for $q \in [m]$ **sample without replacement**:

With prob. $\frac{1}{2} + \frac{j+1}{2n+2}$: $K_q \xleftarrow{\$} \mathcal{U}_{\text{TR}}[j]$

With prob. $\frac{n-j}{2n+2}$: $K_q \xleftarrow{\$} \bigcup_{j'\in[j+1,n]} \mathcal{U}_{\text{TR}}[j']$

Let idx_q s.t. $K_q = \mathcal{U}[\mathsf{idx}_q]$ for all $q \in [m]$

return $\mathsf{enc}(\mathsf{CRS}, \{K_q, \mathsf{idx}_q\}_{q\in[m]})$

$\mathsf{enc}(\mathsf{CRS}, \{K_q, \mathsf{idx}_q\}_{q\in[m]}; v)$

$v \xleftarrow{\$} \mathbb{Z}_p$

for $q \in [m]$ **do**

$Z_q \leftarrow v \cdot \sum_{i\in[n]} [c_i s_i]_1 \cdot K_q.A_i^{(q)}$

$/\!/ \equiv \sum_{i\in[n]} \left[v c_i (s_i + r_i \alpha_i^{(q)})\right]_1$

$k \leftarrow v \cdot [s]_T$

$c \leftarrow \left([v]_2, \{Z_q, \mathsf{idx}_q\}_{q\in[m]}\right)$

return (k, c)

Figure 9: The Enc and TrEnc algorithms, making use of the enc subroutine.

The remaining algorithms. Let $\mathcal{S}_{\mathcal{U},m}$ denote the randomness space for sampling m values uniformly without replacement from $\mathcal{U}$, and let $\mathcal{H}\colon \{0,1\}^\lambda \to \mathcal{S}_{\mathcal{U},m}$ be a hash function modeled as random oracle.

Encryption (Enc and TrEnc). Both Enc and TrEnc work in roughly the same manner: First, a set of m keys $((K_1, \mathsf{idx}_1), \ldots, (K_m, \mathsf{idx}_m))$ is chosen from the preprocessing $\mathcal{U}$, where idx_q denotes the index of K_q in $\mathcal{U}$, and then the subroutine enc (Fig. 9) is used to encrypt to the chosen keys. Enc and TrEnc only differ in *how* the keys are selected:

- $\mathsf{Enc}(\mathsf{pk})$. For normal encryption, keys are selected uniformly random from $\mathcal{U}$ by sampling a pre-image pre and selecting keys according to $\mathcal{H}(\mathsf{pre})$.
- $\mathsf{TrEnc}(\mathsf{tk}, j)$. In $\mathsf{TrEnc}(\mathsf{tk}, j)$, the tracer selects keys by an appropriate distribution over $\bigcup_{j'\in[j,n]} \mathcal{U}_{\text{TR}}[j']$, i.e., from all preprocessing keys where at least the first j parties are revoked. Details can be found in Fig. 9.

$d_i \leftarrow \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)$. Parse c as $([v]_2, \{Z_q, \mathsf{idx}_q\}_{q\in[m]})$ and let $K_q := \mathcal{U}[\mathsf{idx}_q]$. Find a key in $K_1, \ldots, K_m$ where party i is not revoked by checking if

$$(1/\alpha_i^{(q)}) \cdot K_q.A_i^{(q)} \circ [1/c_i]_2 = K_q.[\beta]_1 \circ [1]_2 .$$

The above holds because if party i is unrevoked in K_q, then $K_q.A_i^{(q)} = \left[c_i \alpha_i^{(q)} \cdot K_q.\beta\right]_1$. If no such q exists, output $\perp$. Else, let q be such an index, let $[\beta]_1 = K_q.[\beta]_1$, let $B_i^{(q)} = K_q.B_i^{(q)}$, and output

$$d_i \leftarrow \frac{Z_q \circ [1/c_i]_2}{\left(\alpha_i^{(q)} \cdot [\beta]_1 \circ [v]_2\right) + \left(B_i^{(q)} + \sum_{j\in[n]\setminus\{i\}} [s_j c_j / c_i]_1\right) \circ [v]_2} \equiv [s_i v]_T .$$

$d_i \leftarrow \mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$. Parse r_{Enc} as $(v \in \mathbb{Z}_p, \ldots)$, and output the simulated partial decryption as $d_i \leftarrow v \cdot [s_i c_i]_1 \circ [1/c_i]_2 \equiv [s_i v]_T$.

$k \leftarrow \mathsf{Comb}(\mathsf{pk}, c, S, \{d_i\}_{i\in S})$. Given partial decryptions from parties in $S \subseteq [n]$ with $|S| \geq t$, compute and output $k \leftarrow \sum_{i\in S} \lambda_{i,S}^{(0)} \cdot d_i \equiv [vs]_T$.

Remark 6.1 To avoid complexity in our security analysis, we do not explicitly model the zero-knowledge proofs used in the Setup protocol for TrCom, KGen, and Prep. Since all of these prove knowledge of the randomness used to execute the algorithm, we instead assume in security games that adversaries who control parties that issue such a proof submit the randomness directly to the challenger, who executes the algorithm for them. This avoids boilerplate handling of NIZK-PoKs in proofs through simulation and extraction. Under the hood, we assume the existence of (straight-line) extractable NIZK PoKs for these algorithms. The same remark applies to our security analysis of the transform presented in Section 5. Note that this is a standard technique, and can be observed, for example, in [48].

6.1 Security Proof for our TPLBE-PD scheme

Lemmas 6.2 through 6.6 show that Construction 2 satisfies all properties of TPLBE-PD. Correctness is evident from the construction: if a party decrypts under a preprocessing key where it is not revoked, it will produce $[s_i v]_T$, and t such partial decryptions can be interpolated to recover $k = [sv]_T$.

Lemma 6.2 *For any $\varepsilon \in 1/\mathsf{poly}(\lambda)$ with $\varepsilon \in [0, 1/2]$ and $m = \lambda$, Construction 2 achieves simulability of partial decryptions in the random oracle model.*

Proof Sketch. If at least one of the preprocessing keys chosen for a given ciphertext revokes no parties, simulated partial decryptions from SimPD are clearly identical to real partial decryptions from PDec. Thus, winning GameSimPD amounts to finding a ciphertext, for which a party is revoked in all $m = \lambda$ preprocessing keys. Since preprocessing keys of valid ciphertexts are selected via a random oracle, and half of the preprocessing keys $\mathcal{U}_{\text{Norm}}$ revoke no parties, the probability of generating such a ciphertext is bound by $q/2^\lambda$, where $q \in \mathsf{poly}(\lambda)$ bounds the number of random oracle queries. A full proof is given in Appendix C.1

Lemma 6.3 *For any $\varepsilon \in 1/\mathsf{poly}(\lambda)$ such that $\varepsilon \in [0, 1/2]$ and $m = \lambda$, Construction 2 is (δ, ρ, Q)-indistinguishable for $\delta = \varepsilon/8$, $\rho = t - 1$, and $Q = 2^7 \cdot 3n^2\lambda/\varepsilon^2$ if the commitment scheme Π_{Com} is hiding.*

The full proof can be found in Appendix C.2; we next give a sketch.

Proof Sketch. Let $U := \mathcal{U}_{\text{Norm}} \cup \mathcal{U}_{\text{TR}}[0]$. If the commitment com is hiding, it does not reveal the shuffling permutation (and thus which keys belong to $\mathcal{U}_{\text{TR}}$ and $\mathcal{U}_{\text{Norm}}$). Moreover, all keys in $\mathcal{U}_{\text{Norm}}$ and $\mathcal{U}_{\text{TR}}[0]$ are generated via $\mathsf{Rand}(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i\in[n]}, \emptyset)$ (i.e., they revoke no parties), and are therefore identically distributed. Intuitively, a decoder can thus distinguish Enc and $\mathsf{TrEnc}(0)$ only by the frequency with which certain keys appear in ciphertexts. However, since we consider stateless decoders and sample keys without replacement, the decoder cannot infer the frequency across queries. While it may attempt to gain an advantage by hardcoding specific outputs for certain keys, any such advantage is small except with negligible probability because it cannot know which keys belong to $\mathcal{U}_{\text{Norm}}$ and which to $\mathcal{U}_{\text{TR}}[0]$.

We formalize this via reduction to a statistical problem KEYS-IND (c.f. Appendix D). In this game, an adversary obtains the integers $[L \cdot d]$ and outputs a decoder D that, on input a query $q \in ([Ld] \cup \{\perp\})^\lambda$, outputs a bit. The challenger samples a random subset $U_0 \subseteq [Ld]$ of size $(L-1)d$ and sets $U_1 = [Ld] \setminus U_0$. It then defines distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ over $(U_0 \cup U_1 \cup \{\perp\})^\lambda$ such that $\mathcal{D}_0$ is equivalent to the distribution of Enc over $(\mathcal{U}_{\text{Norm}} \cup \mathcal{U}_{\text{TR}}[0] \cup \mathcal{U}_{\text{TR}}[\geq 1])^\lambda$, and $\mathcal{D}_1$ matches the key distribution induced by $\mathsf{TrEnc}(0)$. The adversary wins if its decoder distinguishes samples from $\mathcal{D}_0$ and $\mathcal{D}_1$ with advantage δ across Q trials. Since the adversary outputs its decoder before U_0 and U_1 are sampled, it has no information about the sets, implying an expected advantage of 0. We prove that deviations from this expectation exceed δ only with negligible probability for the lemma's parameters.

In the reduction from the indistinguishability game to KEYS-IND, the preprocessing keys $\mathcal{U}$ are simulated honestly, except that the reduction does not know which keys for $\mathcal{R}_0$ belong to $\mathcal{U}_{\text{Norm}}$ and which to $\mathcal{U}_{\text{TR}}[0]$. It collects all keys for $\mathcal{R}_0$ in the array U of size Ld. Given a decoder D by the adversary against IndEmp, the reduction constructs a decoder D' for KEYS-IND as follows on input $q \in (v_1, \ldots, v_\lambda)$: For each $i \in [\lambda]$, if $v_i = \perp$, the reduction chooses a random key $K_i \xleftarrow{\$} \mathcal{U}_{\text{TR}}[\geq 1]$. Otherwise, it chooses $K_i \leftarrow U[v_i]$. It then encrypts normally using enc under preprocessing keys $(K_1, \ldots, K_\lambda)$, obtaining (k_0, c), and samples $k_1 \xleftarrow{\$} \mathcal{K}$. It samples a random bit b, queries $D(k_b, c)$, supplies simulated partial decryptions as requested, and receives a guess b'. Finally, D' outputs 1, whenever $b' = b$. Clearly, when queries are sampled from $\mathcal{D}_0$, the outputs of D' count the number of correct guesses of D for Enc ciphertexts. If fed

with queries from $\mathcal{D}_1$, the outputs of D' count the number of correct guesses of D for $\mathsf{TrEnc}(0)$ ciphertexts. Hence, the reduction perfectly simulates the IndEmp game, and wins $\mathsf{KEYS\text{-}IND}$ whenever the internal adversary wins IndEmp.

Lemma 6.4 *For any $\varepsilon \in 1/\mathsf{poly}(\lambda)$ such that $\varepsilon \in [0, 1/2]$ and $m = \lambda$, Construction 2 is (δ, ρ, Q)-index hiding for $\delta = \varepsilon/(4n)$, $\rho = t - 1$, and $Q = 2^7 \cdot 3n^2\lambda/\varepsilon^2$ under the XDH assumption (Asmpt. 1) if the commitment scheme Π_{Com} is hiding.*

Proof Sketch. The index hiding proof is similar to the indistinguishability proof and again reduces to $\mathsf{KEYS\text{-}IND}$ for appropriate parameters. However, we must first show that $\mathcal{U}_{\mathrm{TR}}[i^* - 1] \approx_c \mathcal{U}_{\mathrm{TR}}[i^*]$. Intuitively, without knowledge of $\{\alpha_i^{(q)}\}_{q\in[m]}$, an adversary given $[\beta]_1$ and $\{[r_i\alpha_i^{(q)}]_1\}_{q\in[m]}$ cannot know whether party i was rerandomized with $r_i = \beta$ (unrevoked) or with independent randomness $r_i \xleftarrow{\$} \mathbb{Z}_p^*$ (revoked) in a preprocessing key. We prove this by a reduction to XDH, showing that without knowing sk_{i^*}, an adversary cannot distinguish whether the set $\mathcal{R}_{i^*}$ or the set $\mathcal{R}_{i^*-1}$ is revoked for a given preprocessing key. The full proof can be found in Appendix C.3.

Lemma 6.5 *For any $\varepsilon \in 1/\mathsf{poly}(\lambda)$ such that $\varepsilon \in [0, 1/2]$, Construction 2 is $\mathsf{SIND\text{-}CPA}$-secure under the n-EBDDH assumption (Asmpt. 2), where n is the number of parties if the commitment scheme Π_{Com} is binding.*

Proof of Lemma 6.5. First, we argue that due to the binding property of the commitment scheme, the adversary (who controls the tracer) cannot use different r_{Prep} and $r_{\mathsf{Prep}}{}'$ as randomness for the TrCom and Prep phases, which both open to the commitment com. Henceforth, we assume that the adversary sends the random coins r_{Prep} to the challenger in the TrCom phase, and the challenger uses these coins to generate the preprocessing in the Prep phase (see Remark 6.1).

We prove $\mathsf{SIND\text{-}CPA}$ security by reduction to the n-EBDDH assumption, where n is the number of parties. Let $\mathcal{A}$ be a PPT adversary against $\mathsf{SIND\text{-}CPA}$ security of Construction 2. W.l.o.g., assume that $\mathcal{A}$ initially outputs n parties and that the PPT reduction $\mathcal{B}$ is running inside the n-EBDDH experiment, i.e., $\mathcal{B}$ receives the the tuple

$$\mathsf{view} = \left(E, [a]_1, [b]_2, [v]_2, \{[c_i]_1, [abc_i]_1, [vc_i]_1, [1/c_i]_2\}_{i\in[n]}, \{[c_i/c_j]_1, [abc_i/c_j]_1\}_{i\neq j\in[n]}\right),$$

as well as a challenge $T \in \mathbb{G}_T$ such that either $T = [abv]_T$, or $T = [z]_T$ for a random $z \xleftarrow{\$} \mathbb{Z}_p$.

$\mathcal{B}$ runs $\mathcal{A}$ to receive a threshold $0 < t \leq n$ and a set of corrupted parties $\mathsf{Cor} \subseteq [n]$ with $|\mathsf{Cor}| < t$ (otherwise, $\mathcal{B}$ immediately aborts and outputs a random bit to its challenger). W.l.o.g., assume that $|\mathsf{Cor}| = t - 1$. Denote $\mathcal{H} = [n] \setminus \mathsf{Cor}$.

Simulating the CRS. To simulate the CRS, $\mathcal{B}$ samples $\tilde{s_i} \xleftarrow{\$} \mathbb{Z}_p$ for all $i \in \mathsf{Cor}$. Then, it simulates such that $s \equiv a \cdot b$, and $s_i \equiv \tilde{s_i}$, where a and b are the scalars hidden in the n-EBDDH challenge. More formally, $\mathcal{B}$ computes and assembles the CRS as follows (elements from the n-EBDDH tuple are highlighted in gray):

- Use cross terms $\{[c_i]_1, [1/c_i]_2, \{[c_i/c_j]_1\}_{j\neq i\in[n]}\}_{i\in[n]}$ from the n-EBDDH tuple.
- Set $[s]_T \leftarrow [a]_1 \circ [b]_2$ and

$$[s_ic_i]_1 \leftarrow \begin{cases} \tilde{s_i} \cdot [c_i]_1 & \text{for } i \in \mathsf{Cor} \\ \lambda^{(i)}_{0,\mathsf{Cor}\cup\{0\}} \cdot [abc_i]_1 + \sum_{j\in\mathsf{Cor}} \lambda^{(i)}_{j,\mathsf{Cor}\cup\{0\}} \tilde{s_j} \cdot [c_i]_1 & \text{for } i \in \mathcal{H}. \end{cases}$$

- For all $i \neq j \in [n]$, set

$$[s_ic_i/c_j]_1 \leftarrow \begin{cases} \tilde{s_i} \cdot [c_i/c_j]_1 & \text{for } i \in \mathsf{Cor} \\ \lambda^{(i)}_{0,\mathsf{Cor}\cup\{0\}} [abc_i/c_j]_1 + \sum_{\ell\in\mathsf{Cor}} \lambda^{(i)}_{\ell,\mathsf{Cor}\cup\{0\}} \tilde{s_\ell} \cdot [c_i/c_j]_1 & \text{for } i \in \mathcal{H}, \end{cases}$$

  and compute $\sum_{j\in[n]\setminus\{i\}} [s_jc_j/c_i]_1$ for all $i \in [n]$.

Next, $\mathcal{B}$ forwards the CRS to $\mathcal{A}$, and receives the randomness r_{Prep} from $\mathcal{A}$, which $\mathcal{B}$ parses as $\{\beta^{(u)}, \{r_i^{(u)}\}_{i\in[n]}\}_{u\in[2N]}$. Next, $\mathcal{B}$ simulates key generation on behalf of the honest parties.

Simulating KGen for honest parties. First, $\mathcal{B}$ samples the set of preprocessing keys that will be used in the challenge ciphertext ahead of time: It samples $\mathsf{pre} \xleftarrow{\$} \{0,1\}^\lambda$ and computes $\{u_q\}_{q\in[m]} \leftarrow \mathsf{Samp}([2N]; \mathcal{H}(\mathsf{pre}))$. Then $\mathcal{B}$ samples $\tilde{\alpha}_i^{(q)} \xleftarrow{\$} \mathbb{Z}_p^*$ for all $i \in [n] \setminus \mathsf{Cor}$ and all $q \in [m]$ and simulates key generation such that $\alpha_i^{(q)} \equiv (\tilde{\alpha}_i^{(q)} - s_i)/r_i^{(u_q)}$. In particular, it simulates for all $i \in \mathcal{H}$ as follows, and assembles the result in pk_i, respectively:

- $\left[c_i\alpha_i^{(q)}\right]_1 \leftarrow \frac{\tilde{\alpha}_i^{(q)}}{r_i^{(u_q)}} \cdot [c_i]_1 - \frac{1}{r_i^{(u_q)}} \cdot [s_i c_i]_1$ for all $q \in [m]$. Note that $[s_i c_i]_1$ is not part of the n-EBDDH challenge, but has been simulated as part of the CRS.
- For all $j \in [n] \setminus \{i\}$ and $q \in [m]$: $\left[\alpha_i^{(q)} c_i/c_j\right]_1 \leftarrow \frac{\tilde{\alpha}_i^{(q)}}{r_i^{(u_q)}} [c_i/c_j]_1 - \frac{1}{r_i^{(u_q)}} [s_i c_i/c_j]_1$.

Next, $\mathcal{B}$ forwards $\{\mathsf{pk}_i\}_{i\in[n]}$ to $\mathcal{A}$, and receives as a response the randomness $\{\alpha_i^{(q)}\}_{q\in[m]}$ for key generation on behalf of all corrupted parties $i \in \mathsf{Cor}$, from which $\mathcal{B}$ computes $\{\mathsf{pk}_i\}_{i\in\mathsf{Cor}}$. Then, $\mathcal{B}$ computes the preprocessing $\mathcal{U} \leftarrow \mathsf{Prep}(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i\in[n]}; r_{\mathsf{Prep}})$, using the randomness r_{Prep} it was given by $\mathcal{A}$ during the TrCom phase.

Simulating the challenge (k, c)**.** To simulate the challenge ciphertext, $\mathcal{B}$ simulates encryption under the preprocessing keys $K_{u_1}, \ldots, K_{u_m}$. Recall that $\mathcal{B}$ sampled $\{u_q\}_{q\in[m]}$ according to protocol, but already during simulation of the key generation. Also, recall that $\beta^{(u_q)} = K_{u_q}.\beta$ and $r_i^{(u_q)} = K_{u_q}.r_i$, where $r_i = \beta$ for unrevoked parties. $\mathcal{B}$ embeds the n-EBDDH challenge into the KEM key k by setting $k \leftarrow T$, and for all $q \in [m]$ computes

$$Z_q \leftarrow \sum\nolimits_{i\in\mathsf{Cor}} (\tilde{s}_i + r_i^{(u_q)}\alpha_i^{(q)}) \cdot [c_i v]_1 + \sum\nolimits_{i\in\mathcal{H}} \tilde{\alpha}_i^{(q)} \cdot [c_i v]_1.$$

Since $\tilde{\alpha}_i^{(q)} \equiv s_i + K_{u_q}.r_i \cdot \alpha_i^{(q)}$ for all honest parties $i \in [n] \setminus \mathsf{Cor}$, this is equivalent to

$$Z_q \equiv \sum_{i\in[n]} \left[v c_i (s_i + K_{u_q}.r_i \alpha_i^{(q)})\right]_1.$$

$\mathcal{B}$ assembles $c = ([v]_2, \{Z_q, u_q\}_{q\in[m]})$, and forwards (k, c) to $\mathcal{A}$ simulating the $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ oracle. $\mathcal{B}$ receives a bit b' as a response, which it forwards to its n-EBDDH challenger.

Simulating $\mathcal{O}^{\mathsf{PDec\text{-}Val}}(i, r_{\mathsf{Enc}})$**.** When $\mathcal{A}$ queries $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$, $\mathcal{B}$ parses r_{Enc} as $(v' \in \mathbb{Z}_p, \mathsf{pre} \in \{0,1\}^\lambda)$. $\mathcal{B}$ computes $(k', c') \leftarrow \mathsf{Enc}(\mathsf{pk}, r_{\mathsf{Enc}})$. Denote with $(K'_1, \ldots, K'_\lambda)$ the preprocessing keys used for the ciphertext c'. $\mathcal{B}$ checks whether party i is unrevoked for one of the preprocessing keys, which it can do because it knows the preprocessing randomness r_{Prep}. If this is the case, $\mathcal{B}$ outputs $v' \cdot [s_i]_T$. Otherwise, $\mathcal{B}$ outputs $\perp$, prefectly simulating the $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ oracle.

Analysis. Observe that, if the n-EBDDH challenge $T = [abv]_T$, then $\mathcal{B}$ simulates the SIND-CPA game perfectly for $b = 0$. If, however $T = [z]_T$ for a random z, then $\mathcal{B}$ perfectly simulates the SIND-CPA game for $b = 1$. Whenever the SIND-CPA game would abort, for example because $\mathcal{A}$ outputs some Cor with $|\mathsf{Cor}| \geq t$, $\mathcal{B}$ outputs a random bit. Hence $\mathcal{B}$ guesses the internal bit of the n-EBDDH challenge correctly with the same advantage as $\mathcal{A}$ achieves against SIND-CPA, and we conclude that $\mathsf{Adv}^{\mathsf{SIND\text{-}CPA}}_{\mathcal{A},\mathsf{TPLBE\text{-}PD}}(\lambda) = \mathsf{Adv}^{n\text{-}\mathsf{EBDDH}}_{\mathcal{B},\mathcal{G}}(\lambda)$, which is negligible by assumption. ∎

The message privacy proof follows the techniques of the SIND-CPA proof, except that we now rely on an interactive assumption called (t, n)-Int-EBDDH (Asmpt. 3). It is very similar to the more established n-EBDDH (Asmpt. 2) used to prove SIND-CPA security, except that it allows us to adaptively select a set $\mathcal{I}_{\mathsf{pd}}$ of size $t - 1$, returning $\{[s_i v]_T\}_{i\in\mathcal{I}_{\mathsf{pd}}}$. We use this to simulate SimPD for $\mathcal{I}_{\mathsf{pd}}$ on the challenge ciphertext. We prove that Assumption 3 holds in the GGM and use it to prove Lemma 6.6 in Appendix E and Appendix C.4, respectively.

Assumption 3 ((t, n)-Interactive Extended Bilinear Decisional Diffie-Hellman ((t, n)-Int-EBDDH)). Let $\mathcal{G}$ be an algorithm to generate a pairing ensemble E, let $n, t \in \mathsf{poly}(\lambda)$ and $0 < t \leq n$. The (t, n)-interactive extended bilinear decisional Diffie Hellman problem ((t, n)-Int-EBDDH) is hard for $\mathcal{G}$ if for all PPT adversaries $\mathcal{A}$, the following function is negligible in λ:

$$\mathsf{Adv}^{(t,n)\text{-}\mathsf{Int\text{-}EBDDH}}_{\mathcal{A},\mathcal{G}}(1^\lambda) \coloneqq \Pr\left[\mathsf{Int\text{-}EBDDH}_{\mathcal{A}}(1^\lambda, n, t) = 1\right] - \frac{1}{2},$$

where the Int-EBDDH game is defined in Fig. 10.

Lemma 6.6 *For any $\varepsilon \in 1/\mathsf{poly}(\lambda)$ with $\varepsilon \in [0, 1/2]$, Construction 2 satisfies $(t-1)$-message privacy under the (t, n)-Int-EBDDH assumption, where n is the number of parties, and t the threshold if the commitment scheme Π_{Com} is hiding.*

Game $\mathsf{Int\text{-}EBDDH}_{\mathcal{A}}(1^\lambda, n, t)$

$S \leftarrow \{0, 1, \ldots, t-1\}$

$E \coloneqq (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \circ, [1]_1, [1]_2, p) \xleftarrow{\$} \mathcal{G}(1^\lambda)$

$\forall \ell \in S$: Sample $y_\ell \xleftarrow{\$} \mathbb{Z}_p$

$\forall i \in [n]$: Sample $c_i \xleftarrow{\$} \mathbb{Z}_p^*$

Sample $v, z \xleftarrow{\$} \mathbb{Z}_p$

$$\mathsf{view}_1 \leftarrow \left(\left\{ [c_i]_1, [vc_i]_1, \left[\sum_{\ell \in S} \lambda_{\ell,S}^{(i)} y_\ell c_i \right]_1 \right\}_{i \in [n]}, \left\{ [c_i/c_j]_1, \left[\sum_{\ell \in S} \lambda_{\ell,S}^{(i)} y_\ell c_i / c_j \right]_1 \right\}_{i \neq j \in [n]} \right)$$

$$\mathsf{view}_2 \leftarrow \left([v]_2, \left\{ [1/c_i]_2 \right\}_{i \in [n]} \right)$$

$$\mathsf{view}_T \leftarrow \left([y_0]_T \right)$$

$b \xleftarrow{\$} \{0, 1\}$

if $b = 0$ **then** $\mathsf{chal} \leftarrow [y_0 \cdot v]_T$ **else** $\mathsf{chal} \leftarrow [z]_T$

$(\mathcal{I}, \mathsf{st}) \xleftarrow{\$} \mathcal{A}(1^\lambda, E, \mathsf{view}_1, \mathsf{view}_2, \mathsf{view}_T, \mathsf{chal})$

if $\mathcal{I} \not\subseteq [n] \vee |\mathcal{I}| \geq t$ **then return** 0

$$\mathsf{view}'_T \leftarrow \left\{ \left[v \cdot \sum\nolimits_{\ell \in S} \lambda_{\ell,S}^{(i)} y_\ell \right]_T \right\}_{i \in \mathcal{I}}$$

$b' \xleftarrow{\$} \mathcal{A}(\mathsf{st}, \mathsf{view}'_T)$

return $b \stackrel{?}{=} b'$

Figure 10: The Int-EBDDH experiment.

7 Extending to Fully-Malicious Partial Decryption Oracle

In this section, we show how to extend any TT-KEM that is traceable under a partial decryption oracle for valid ciphertexts $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ to achieve traceability under a fully-malicious partial decryption oracle $\mathcal{O}^{\mathsf{PDec}}$. The fully-malicious partial decryption oracle $\mathcal{O}^{\mathsf{PDec}}(i, c)$ receives a ciphertext c instead of encryption randomness r, and outputs a partial decryption $d_i \leftarrow \mathsf{PDec}(\mathsf{sk}_i, c)$.

In particular, this transform applies to our construction in Section 5. We first introduce definitions for NIZK-PoKs and hard relations.

7.1 NIZK-PoKs and Hard Relations

Definition 7.1 (NIZK-PoK). Let R be an NP-relation of statement-witness pairs $(\chi, \omega) \in \mathsf{R}$ a non-interactive zero-knowledge proof system of knowledge PS_R for R is a tuple of algorithms $\mathsf{PS}_\mathsf{R} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ with the following syntax:

- $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$. Given the security parameter λ, the Setup algorithm outputs public parameters pp, which are implicit input to all subsequent algorithms.
- $\pi \xleftarrow{\$} \mathsf{Prove}(\chi, \omega)$. For a statement-witness pair (χ, ω), the Prove algorithm returns a proof π.
- $1/0 \leftarrow \mathsf{Verify}(\chi, \pi)$. Given a statement χ and a proof π, the Verify algorithm outputs 1, iff π is valid for χ.

A proof system must fulfill *correctness*: For all $(\chi, \omega) \in \mathsf{R}$ it holds that $\mathsf{Verify}(\chi, \mathsf{Prove}(\chi, \omega)) = 1$.

We require a NIZK-PoK to satisfy *simulation extractability*, following the definition of (weak) simulation extractability of [22].

Definition 7.2 (Simulation Extractability). A proof system PS_R for R is weakly simulation-extractable if there exists a PPT simulator $\mathsf{Sim} = (\mathsf{Setup}, \mathsf{Sim}, \mathsf{Ext})$ such that for all PPT adversaries $\mathcal{A}$ there exists a negligible function negl such that

$$|\Pr\left[\mathsf{SimExt}_{\mathcal{A}}^{\mathsf{PS}_\mathsf{R}, \mathsf{Sim}}(1^\lambda) = 1\right] - \frac{1}{2}| \leq \mathsf{negl}(\lambda),$$

where SimExt is defined in Figure 11.

$\mathsf{SimExt}_{\mathcal{A}}^{\mathsf{PS_R},\mathsf{Sim}}(1^\lambda)$	Oracle $\mathcal{O}_b^{\mathsf{prove}}(\chi,\omega)$	Oracle $\mathcal{O}_b^{\mathsf{verify}}(\chi,\pi)$
$b \xleftarrow{\$} \{0,1\}; Q \leftarrow \emptyset$	**if** $(\chi,\omega) \notin \mathsf{R}$ **then return** $\bot$	**if** $b = 0$ **then**
if $b = 0$	**if** $b = 0$	**return** $\mathsf{Verify}(\chi,\pi)$
$\mathsf{pp} \xleftarrow{\$} \mathsf{PS_R.Setup}(1^\lambda)$	$\pi \xleftarrow{\$} \mathsf{PS_R.Prove}(\chi,\omega)$	**else**
else	**else**	**if** $\mathsf{Verify}(\chi,\pi) = 0$ **then return** 0
$(\mathsf{pp},\mathsf{st}_{\mathsf{Sim}}) \xleftarrow{\$} \mathsf{Sim.Setup}(1^\lambda)$	$(\pi,\mathsf{st}_{\mathsf{Sim}}) \xleftarrow{\$} \mathsf{Sim.Sim}(\chi,\mathsf{st}_{\mathsf{Sim}})$	**if** $\chi \in Q$ **then return** 1
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_b^{\mathsf{prove}},\mathcal{O}_b^{\mathsf{verify}}}(1^\lambda,\mathsf{pp})$	$Q \leftarrow Q \cup \{\chi\}$	$(\omega,\mathsf{st}_{\mathsf{Sim}}) \xleftarrow{\$} \mathsf{Sim.Ext}(\chi,\pi,\mathsf{st}_{\mathsf{Sim}})$
return $b' \stackrel{?}{=} b$	**return** π	**if** $(\chi,\omega) \notin \mathsf{R}$ **then return** $\bot$
		return 1

Figure 11: The simulation-extractability game SimExt for NIZK-PoKs.

Definition 7.3 (Hard Relation). A hard relation H is a set of tuples (Y, y) such that

- there exists a PPT sampling algorithm $(Y, y) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$ that outputs a statement-witness pair $(Y, y) \in \mathsf{H}$.
- the relation is decidable in poly-time.
- for all PPT adversaries $\mathcal{A}$ there exists a negligible function negl such that

$$\Pr\big[(Y, \mathcal{A}(1^\lambda, Y)) \in \mathsf{H} \colon (Y, y) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)\big] \leq \mathsf{negl}(\lambda).$$

7.2 Transform

At a high level, our transformation works by attaching to a ciphertext c a NIZK proof of knowledge of either the randomness r_{Enc} used to generate it, or a secret trapdoor known to the tracer. This allows us to extract r_{Enc} for ciphertexts sent to $\mathsf{PDec}(i, c)$ by the adversary, and then use r_{Enc} to simulate partial decryptions by querying $\mathcal{O}^{\mathsf{PDec\text{-}Val}}(i, r_{\mathsf{Enc}})$. During tracing, the tracer can simulate proofs for malformed ciphertexts using the trapdoor.

More formally, given a TT-KEM Π, our transform is based on a hard relation H and NIZK-PoK system $\mathsf{PS_R}$ for the relation

$$\mathsf{R} = \left\{ (\chi, \omega) \colon \begin{array}{c} \chi = (\mathsf{pp}, \mathsf{pk}, \mathsf{ptd}, c) \wedge \\ (\Pi.\mathsf{Enc}(\mathsf{pk}; \omega) = c \vee (\mathsf{ptd}, \mathsf{td}) \in \mathsf{H}) \end{array} \right\}.$$

The transform Π' makes the following modifications to the Setup, Enc, PDec, and Trace algorithms of Π', keeping Comb unchanged.

$\Pi'.\mathsf{Setup}(1^\lambda, n, t, \varepsilon)$**.** First, the proof system $\mathsf{PS_R}$ is set up, generating $\mathsf{pp}_{\mathsf{PS}}$. Then, run $\Pi.\mathsf{Setup}(1^\lambda, n, t, \varepsilon/2)$ to generate the tuple $(\mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{tk})$. At the end of the setup protocol, the tracer samples a tuple $(\mathsf{ptd}, \mathsf{td})$ of a public commitment ptd and a trapdoor td as $(\mathsf{ptd}, \mathsf{td}) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$. Set $\mathsf{pp}' \leftarrow (\mathsf{pp}, \mathsf{pp}_{\mathsf{PS_R}})$, $\mathsf{pk}' = (\mathsf{pk}, \mathsf{ptd})$ and $\mathsf{tk}' = (\mathsf{tk}, \mathsf{td})$. The output of $\Pi'.\mathsf{Setup}$ is $(\mathsf{pp}', \mathsf{pk}', \{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{tk}')$

$\Pi'.\mathsf{Enc}(\mathsf{pk}')$. Sample $r_{\mathsf{Enc}} \xleftarrow{\$} \Pi.\mathcal{R}$, and compute $(k, c) \leftarrow \Pi.\mathsf{Enc}(\mathsf{pk}; r_{\mathsf{Enc}})$. Assemble $\chi \coloneqq (\mathsf{pp}, \mathsf{pk}, \mathsf{ptd}, c)$ and $\omega \coloneqq r_{\mathsf{Enc}}$ and compute $\pi \xleftarrow{\$} \mathsf{PS.Prove}(\chi, \omega)$. Output k and $c' = (c, \pi)$.

$\Pi'.\mathsf{PDec}(\mathsf{pk}', \mathsf{sk}_i, c')$**.** First, parse c' as (c, π), and check whether $\mathsf{PS.Verify}((\mathsf{pp}, \mathsf{pk}, \mathsf{ptd}, c), \pi) = 1$. If verification fails, output $\bot$. Otherwise, output $d_i \leftarrow \Pi.\mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)$.

$\Pi'.\mathsf{Trace}^{D'}(\mathsf{pk}', \mathsf{tk}', \varepsilon)$**.** Parse tk' as $(\mathsf{tk}, \mathsf{td})$. Then, construct a decoder D that does the following on input (k, c):

- Assemble $\chi = (\mathsf{pp}, \mathsf{pk}, \mathsf{ptd}, c)$ and generate a proof $\pi \xleftarrow{\$} \mathsf{PS.Prove}(\chi, \mathsf{td})$ using the trapdoor td, and assemble $c' = (c, \pi)$.
- Query $(\mathcal{I}_{\mathsf{pd}}, \mathsf{st}) \xleftarrow{\$} D'(k, c')$, and outputs $(\mathcal{I}_{\mathsf{pd}}, \mathsf{st})$.
- On input $(\mathsf{st}, \{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}})$, run $b \xleftarrow{\$} D'(\mathsf{st}, \{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}})$ and output the bit b.

Finally, run and output $\mathcal{T}' \xleftarrow{\$} \Pi.\mathsf{Trace}^D(\mathsf{pk}, \mathsf{tk}, \varepsilon/2)$.

Theorem 7.4 *If* H *is a hard relation,* PS_{R} *is simulation-extractable, and* Π *is traceable against* ρ *partial insider under* $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ *with limit* ε*, then* Π' *is traceable against* ρ *partial insiders under* $\mathcal{O}^{\mathsf{PDec}}$ *for limit* ε*. If* Π *is* SIND-CPA *secure and* PS_{R} *is zero-knowledge, then* Π' *is also* SIND-CPA *secure.*

Remark 7.5 Our generic transformation above relies on a NIZK-PoK system capable of proving statements about the encryption circuit of the base scheme. This creates a modeling tension when the encryption algorithm of the base scheme invokes a random oracle, as is the case for our construction.

We follow the standard approach of first analyzing the base scheme in the random-oracle model and then treating its security in a concrete instantiation of the random oracle as a black-box assumption when applying the generic transformation. Once the random oracle is instantiated by a concrete hash function, Enc can be viewed as an ordinary circuit, and the NIZK-PoK can be applied to this circuit. The generic transform can therefore also be applied to schemes that use a random oracle during encryption, with the caveat that the base scheme is only proven secure in the random oracle model, while its security is then *assumed* when instantiating the transform. Note that this pattern arises elsewhere in the literature. For example, to construct blind signatures, Fuchbauer and Wolf [37] instantiate a NIZK over Schnorr signatures, despite the fact that the Schnorr signer evaluates a random oracle, by generically assuming unforgeability of Schnorr when instantiated with a concrete hash function prior to the NIZK steps in their analysis. An analogous issue arises in recursive SNARKs in the random oracle model: Recursive verification requires proving over the verifier circuits, which include Fiat-Shamir random oracle evaluations. Prior work either, much like our approach, instantiates the random oracle and assumes that the resulting scheme preserves the required security properties [28], or introduces alternative oracle models specifically designed to formalize such non-blackbox use [26, 25].

Proof Sketch. We prove traceability of Π' through a series of game hops, ending with a reduction to traceability of Π. Let Game_0 be $\mathsf{Game\text{-}Tr}$ instantiated with Π' and a fully-malicious partial decryption oracle $\mathcal{O}^{\mathsf{PDec}}$ for parameters 1^λ, ε and ρ. We define the following game hops:

- Game_1. First, we use the simulator Sim to instantiate the proof system PS as $(\mathsf{pp}_{\mathsf{PS}_{\mathsf{R}}}, \mathsf{st}_{\mathsf{Sim}}) \xleftarrow{\$} \mathsf{Sim.Setup}(1^\lambda)$ during $\Pi'.\mathsf{Setup}$.

  We modify the $\mathcal{O}^{\mathsf{PDec}}(i,(c,\pi))$ oracle to extract a witness ω after verifying the proof π. For this, we set $\chi = (\mathsf{pp}, \mathsf{pk}, \mathsf{ptd}, c)$ as $\omega \xleftarrow{\$} \mathsf{Sim.Ext}(\chi, \pi, \mathsf{st}_{\mathsf{Sim}})$. We let the oracle output $\perp$, if $(\chi, \omega) \notin \mathsf{R}$. By the simulation-extractability of Π, this modification increases the adversary's advantage against $\mathsf{Game\text{-}Tr}$ by at most a negligible amount.

- Game_2. Next, we argue that the extracted witness ω in $\mathcal{O}^{\mathsf{PDec}}$ cannot be a solution to the hard relation H. For this, we add an additional abort condition to $\mathcal{O}^{\mathsf{PDec}}$, returning $\perp$ if $(\mathsf{ptd}, \omega) \in \mathsf{H}$ for the extracted witness ω. We prove indistinguishability of Game_1 and Game_2 by a reduction to the hardness of H. The reduction initially receives as input some ptd, and simulates either tracing game to its internal adversary. If at some point, the additional abort condition in $\mathcal{O}^{\mathsf{PDec}}$ is hit, then the reduction outputs ω and breaks the hardness of H, because it holds that $(\mathsf{ptd}, \omega) \in \mathsf{H}$.

We conclude with a reduction $\mathcal{B}$ to traceability of Π. Crucially, the reduction must simulate $\mathcal{O}^{\mathsf{PDec}}$ as defined under Game_2 (henceforth, $\mathcal{O}_2^{\mathsf{PDec}}$), but can only make use of a partial decryption oracle for valid ciphertexts $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ offered by its challenger. Upon close inspection of $\mathcal{O}_2^{\mathsf{PDec}}(i,(c,\pi))$, observe that $\mathcal{B}$ must return $\perp$ any of the following conditions hold:

- $\mathsf{Verify}((\mathsf{pp}, \mathsf{pk}, \mathsf{ptd}, c), \pi) = 0$
- $((\mathsf{pp}, \mathsf{pk}, \mathsf{ptd}, c), \omega) \notin \mathsf{R}$
- $(\mathsf{ptd}, \omega) \in \mathsf{H}$

Note that the second and third conditions together imply that $\perp$ is returned unless $\Pi.\mathsf{Enc}(\mathsf{pk}; \omega) = c$. In this case, however, $\mathcal{B}$ can simply query its challenger for $\mathcal{O}^{\mathsf{PDec\text{-}Val}}(i, r_{\mathsf{Enc}} := \omega)$ to obtain a partial decryption d_i on c without knowing sk_i, and thus simulate $\mathcal{O}_2^{\mathsf{PDec}}$ perfectly.

In the end, the reduction constructs D from D' as mandated by Game_2. Observe that, whenever D' is good, then so is the decoder D produced by $\mathcal{B}$, and since $\Pi'.\mathsf{Trace}$ determines $\mathcal{T}'$ by forwarding the output of $\Pi.\mathsf{Trace}$, we conclude that $\mathcal{B}$ wins $\mathsf{Game\text{-}Tr}$ against Π with $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ with non-negligible advantage, if $\mathcal{A}$ wins $\mathsf{Game\text{-}Tr}$ against Π with $\mathcal{O}^{\mathsf{PDec}}$ with non-negligible advantage.

We note that the formal proof is significantly more involved. At a high level, there are two challenges beyond what is discussed above: First, a PPT reduction cannot efficiently compute $\mathsf{GoodDec}$, and thus

cannot directly show that $\Pr[\mathsf{GoodDec}]$ does not change significantly between Game_0 and Game_1, as it cannot output bit to the simulation-extractability challenger dependent on whether $\mathsf{GoodDec} = 1$. Second, the simulator Sim for Π could be stateful, and update its state after every simulation and extraction query. Hence, when reducing from Game_2 to traceability of Π, our reduction cannot simply construct D by querying D' on a simulated proof, as the resulting decoder would no longer be stateless. We next show how to overcome these challenges in the proof.

7.3 A formal Proof of Theorem 7.4

Proof of Theorem 7.4. Let $\varepsilon \in 1/\mathsf{poly}(\lambda)$. Before giving a proof outline, we must first define the following three experiments, each of which tests the decoder D' output by $\mathcal{A}$ within $\mathsf{Game\text{-}Tr}$. We will use the experiments to argue the quality of decoders across game hops.

Experiment $\mathsf{test\text{-}real}$**:** Initialize $ctr \leftarrow 0$ and repeat the following $Q = 128\lambda/\varepsilon^2$ times:

- Sample $((k_0, c); r_{\mathsf{Enc}}) \xleftarrow{\$} \Pi.\mathsf{Enc}(\mathsf{pk})$ and $k_1 \xleftarrow{\$} \mathcal{K}$.
- Let $\chi = (\mathsf{pp}, \mathsf{pk}, \mathsf{ptd}, c)$ and compute $\pi \xleftarrow{\$} \mathsf{PS.Prove}(\chi, r_{\mathsf{Enc}})$
- Sample $b \xleftarrow{\$} \{0,1\}$.
- Run $(\mathcal{I}_{\mathsf{pd}}, \mathsf{st}) \xleftarrow{\$} D'(k_b, (c, \pi))$. If $\mathcal{I}_{\mathsf{pd}} \subseteq [n]$ and $|\mathcal{I}_{\mathsf{pd}}| \leq \rho$, compute $d_i \xleftarrow{\$} \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)$ for all $i \in \mathcal{I}_{\mathsf{pd}}$, and let $b' \xleftarrow{\$} D'(\mathsf{st}, \{d_i\}_{i \in \mathcal{I}_{\mathsf{pd}}})$. Otherwise let $b' = 0$.
- If $b = b'$, increment $ctr \leftarrow ctr + 1$

Finally, let the event $\mathsf{test\text{-}real} = 1$, iff $ctr \geq Q \cdot (1/2 + 7\varepsilon/8)$.

Experiment $\mathsf{test\text{-}td}$**:** The experiment is identical to $\mathsf{test\text{-}real}$, except that in the second step, we compute $\pi \xleftarrow{\$} \mathsf{PS.Prove}(\chi, \mathsf{td})$ using the trapdoor. Let the event $\mathsf{test\text{-}td} = 1$, iff $ctr \geq Q \cdot (1/2 + 6\varepsilon/8)$ in the second experiment.

Experiment $\mathsf{test\text{-}td'}$**:** The experiment is identical to $\mathsf{test\text{-}td}$, except that we set $\mathsf{test\text{-}td'} = 1$, iff $ctr \geq Q \cdot (1/2 + 5\varepsilon/8)$.

Let Game_0 be $\mathsf{Game\text{-}Tr}$ instantiated with Π' and a fully-malicious partial decryption oracle $\mathcal{O}^{\mathsf{PDec}}$ for parameters 1^λ, ε and ρ. For an event E, we denote with E_i the event as it is defined with respect to Game_i.

7.3.1 Proof Outline

Overall, we want to show that for all PPT adversaries $\mathcal{A}$ running in the traceability game of Π', it holds that $\Pr\left[\mathsf{GoodTr}_0^{\mathcal{A}}\right] \geq \Pr\left[\mathsf{GoodDec}_0^{\mathcal{A}}\right] - \mathsf{negl}(\lambda)$, and $\Pr\left[\mathsf{BadTr}_0^{\mathcal{A}}\right] \leq \mathsf{negl}(\lambda)$. On a very high level, our proof will work by making modifications to the traceability game instantiated with Π', up to a point, where we can make a reduction $\mathcal{B}$ to the traceability of Π. If we can then bound the probability that our reduction $\mathcal{B}$ produces a good decoder against Π to be at least $\Pr\left[\mathsf{GoodDec}_0^{\mathcal{A}}\right] - \mathsf{negl}(\lambda)$, then traceability of Π gives us that

$$\Pr\left[\mathsf{GoodTr}_0^{\mathcal{A}}\right] \geq \Pr\left[\mathsf{GoodTr}^{\mathcal{B}}\right] - \mathsf{negl}(\lambda) \geq \Pr\left[\mathsf{GoodDec}^{\mathcal{B}}\right] - \mathsf{negl}'(\lambda) \geq \Pr\left[\mathsf{GoodDec}_0^{\mathcal{A}}\right] - \mathsf{negl}''(\lambda),$$

for appropriate negligible functions.

For this, we will introduce two game hops. In Game_1, we instantiate the proof system using the simulator and extract all proofs submitted to the $\mathcal{O}^{\mathsf{PDec}}$ oracle. We also replace the proofs generated in the iterations of the $\mathsf{test\text{-}real}$ and $\mathsf{test\text{-}td}$ experiments with simulated ones. Game_1 and Game_0 are computationally indistinguishable by simulation extractability of PS.

In Game_2, we add an additional abort condition to the $\mathcal{O}^{\mathsf{PDec}}$ oracle, aborting if extraction fails to produce the encryption randomness, and instead outputs the simulation trapdoor. We reduce this to the hardness of H.

Then, we prove that certain events imply other events, except with negligible probability in some games. For this, observe that events are designed with decreasing *success thresholds*: $\mathsf{GoodDec}^{\mathcal{A}}$ holds if the decoder produced by $\mathcal{A}$ has at least advantage $1/2 + \varepsilon$, $\mathsf{test\text{-}real}$ holds if the test succeeds in at least a fraction $1/2 + 7\varepsilon/8$ of the repetition. For $\mathsf{test\text{-}td}$, the threshold is $1/2 + 6\varepsilon/8$, for $\mathsf{test\text{-}td'}$ it is $1/2 + 5\varepsilon/8$, and finally, $\mathsf{GoodDec}^{\mathcal{B}}$ holds if the decoder produced by the reduction has a distinguishing advantage of at least $1/2 + \varepsilon/2$. We give an overview of all the implications we prove and use in Figure 12.

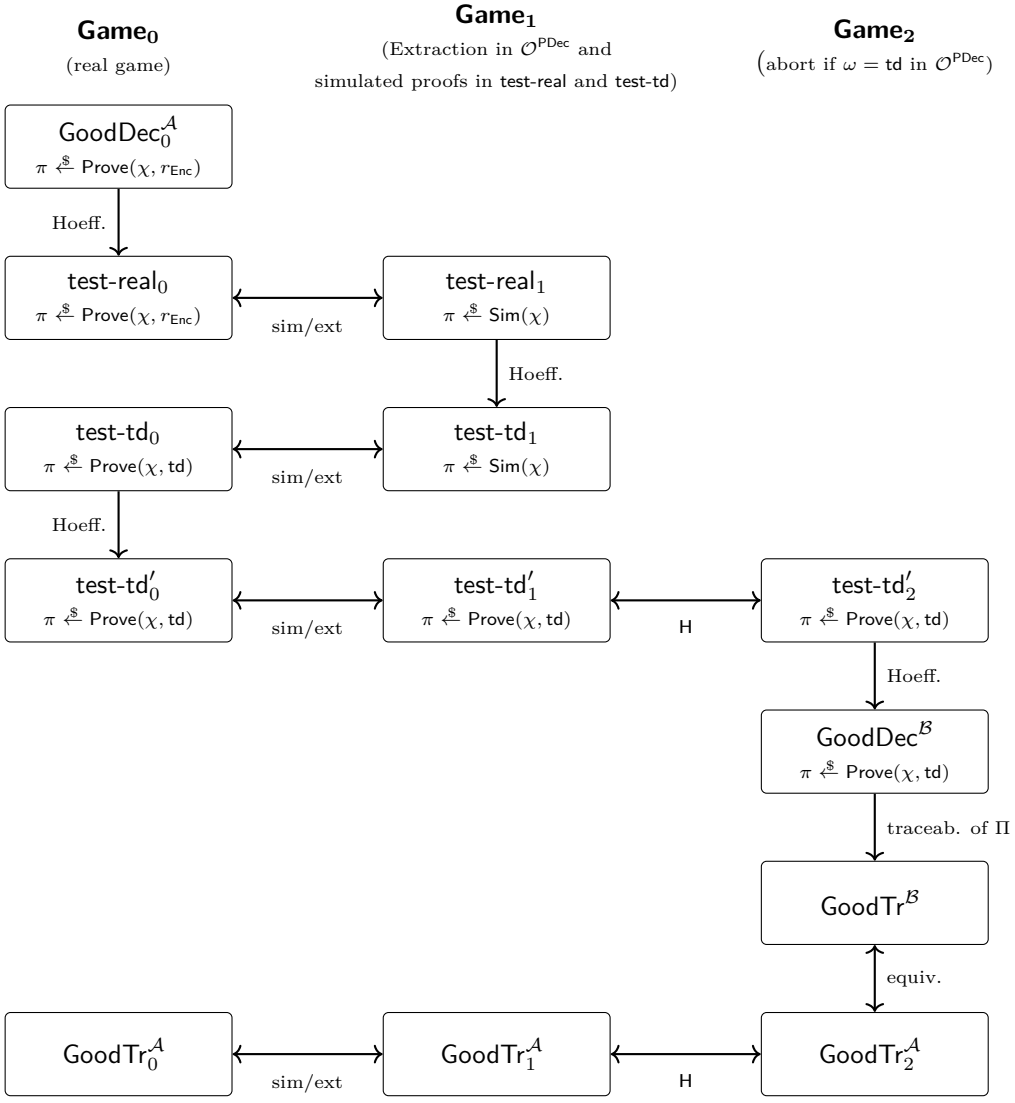

Figure 12: An overview of the chain of implications showing that $\mathsf{GoodDec}^{\mathcal{A}}$ implies $\mathsf{GoodTr}^{\mathcal{A}}$, except with negligible probability. In each box that corresponds to an experiment on a decoder, we write how the proof π is computed, which is attached to the ciphertext c before it is forwarded to the decoder D'. Note that we can apply Hoeffding's inequality to show that event A implies event B except with negligible probability only if the experiments underlying A and B are identical, and if A requires a noticeably higher success threshold than B. Specifically, this means that both experiments must compute the proof π in the same manner.

Intuition behind the use of experiments. First, observe that we can make the reduction $\mathcal{B}$ to traceability of Π under $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ only from Game_2, because otherwise, the reduction would not be able to simulate the $\mathcal{O}^{\mathsf{PDec}}$ oracle. Hence, we need to show for any PPT adversary $\mathcal{A}$, that $\mathcal{B}$ produces a good decoder against its challenger with a probability $\Pr\left[\mathsf{GoodDec}^{\mathcal{B}}\right]$ that is not significantly less than $\Pr\left[\mathsf{GoodDec}_0^{\mathcal{A}}\right]$, i.e., the probability with which $\mathcal{A}$ produces a good decoder against the real traceability game of Π' under $\mathcal{O}^{\mathsf{PDec}}$.

Ideally, we would like to show that $\mathsf{GoodDec}_0^{\mathcal{A}}$ implies $\mathsf{GoodDec}_2^{\mathcal{A}}$, except with negligible probability. We cannot prove this, since we cannot reduce to the simulation extractability of PS_R. This is because the event $\mathsf{GoodDec}$ is not computable by the reduction, as it depends on the *real success probability* of the decoder D', which is not efficiently computable.[6]

Hence, we introduce the event $\mathsf{test\text{-}real}$, which tests the decoder in a manner identical to $\mathsf{GoodDec}_0^{\mathcal{A}}$ (i.e., it generates proofs for D' using $\mathsf{Prove}(\chi, r_{\mathsf{Enc}})$), but is efficiently computable and has a success threshold of just $1/2 + 7\varepsilon/8$. This allows us to prove that $\mathsf{GoodDec}_0^{\mathcal{A}}$ implies $\mathsf{test\text{-}real}_0$, except with negligible probability by a Hoeffding inequality.

Next, observe that the reduction $\mathcal{B}$ from Game_2 to traceability of Π constructs its decoder D as mandated by $\Pi'.\mathsf{Trace}$. In particular, it uses the trapdoor td to attach a proof π to a given ciphertext c as $\pi \xleftarrow{\$} \mathsf{Prove}(\chi, \mathsf{td})$, and then querying D' on (c, π). Intuitively, these should be indistinguishable to D' because of the zero-knowledge property of the proof system. Note, however, that $\mathsf{test\text{-}real}$ computes the proof using r_{Enc} as witness, hence the experiments are not *identical* yet, and we cannot apply a Hoeffding inequality directly.

Instead, we introduce the experiment $\mathsf{test\text{-}td}$, which also uses the trapdoor td to compute the proof π when testing the decoder. Then, in Game_1, having replaced the proofs generated in $\mathsf{test\text{-}real}$ *and* $\mathsf{test\text{-}td}$ with simulated proofs, we can prove that $\mathsf{test\text{-}real}_1$ implies $\mathsf{test\text{-}td}_1$, since now both experiments are identical.[7]

Unfortunately, we still cannot prove that, once we are in Game_2, $\mathsf{test\text{-}td}_2$ implies $\mathsf{GoodDec}^{\mathcal{B}}$, except with negligible probability. This is because now $\mathsf{test\text{-}td}_2$ generates simulated proofs, while $\mathsf{GoodDec}^{\mathcal{B}}$ generates proofs to D' using the trapdoor. To solve this, we introduce the third experiment $\mathsf{test\text{-}td}'$, which is initially identical to $\mathsf{test\text{-}td}$ under Game_0, except that, again, it has a lower threshold $1/2 + 5\varepsilon/8$. This allows us to prove by a Hoeffding bound that $\mathsf{test\text{-}td}$ implies $\mathsf{test\text{-}td}'$, except with negligible probability. Unlike $\mathsf{test\text{-}td}$, however, we do not simulate proofs in $\mathsf{test\text{-}td}'_1$ and $\mathsf{test\text{-}td}'_2$ when moving to Game_1 and Game_2, which ensures that the experiments underpinning $\mathsf{test\text{-}td}'_2$ and $\mathsf{GoodDec}^{\mathcal{B}}$ are identical in Game_2, except that $\mathsf{test\text{-}td}'_2$ has threshold $1/2 + 5\varepsilon/8$ and $\mathsf{GoodDec}^{\mathcal{B}}$ only $1/2 + \varepsilon/2$. As a result, we can use Hoeffding's inequality to show that $\mathsf{test\text{-}td}'_2$ implies $\mathsf{GoodDec}^{\mathcal{B}}$, except with negligible probability, and complete our line of implications from $\mathsf{GoodDec}_0^{\mathcal{A}}$ to $\mathsf{GoodDec}^{\mathcal{B}}$.

Additionally, we show that $\mathsf{GoodTr}_0^{\mathcal{A}}$, $\mathsf{GoodTr}_1^{\mathcal{A}}$, and $\mathsf{GoodTr}_2^{\mathcal{A}}$, as well as $\mathsf{BadTr}_0^{\mathcal{A}}$, $\mathsf{BadTr}_1^{\mathcal{A}}$, and $\mathsf{BadTr}_2^{\mathcal{A}}$ are computationally close, which can be easily proven through reductions to simulation extractability of PS_R and hardness of H, because these events can be efficiently computed by a PPT reduction. An overview of the line of implications outlined above is given in Figure 12.

7.3.2 Formalizing Figure 12

We continue with the formal proof outlined above. First, we argue that in Game_0, the following claim holds:

Claim 7.6 For all PPT adversaries $\mathcal{A}$ running in Game_0 it holds that

$$\Pr[\mathsf{GoodDec}_0] \leq \Pr[\mathsf{test\text{-}real}_0] + \mathsf{negl}(\lambda) \quad \text{and} \quad \Pr[\mathsf{test\text{-}td}] \leq \Pr\left[\mathsf{test\text{-}td}'_0\right] + \mathsf{negl}(\lambda).$$

Proof. The claim follows a simple Hoeffding inequality. ∎

Given the above experiments, we make the following modification to the traceability game in a game hop.

$\underline{\mathsf{Game}_1}$: In Game_1, we use the simulator Sim of the proof system PS to instantiate the setup $(\mathsf{pp}_{\mathsf{PS}_\mathsf{R}}, \mathsf{st}_{\mathsf{Sim}}) \xleftarrow{\$} \mathsf{Sim.Setup}(1^\lambda)$ during $\Pi'.\mathsf{Setup}$. We modify the $\mathcal{O}^{\mathsf{PDec}}(i, (c, \pi))$ oracle to extract a witness ω after verifying

[6] As an example, a distinguishing adversary could always produce a decoder D' which has success probability exactly $1/2 + \varepsilon$ in Game_0, but produce D' that has success probability $1/2 + \varepsilon - \mathsf{negl}(\lambda)$ in Game_1, and thus $\mathsf{GoodDec}_0^{\mathcal{A}} = 1$, but $\mathsf{GoodDec}_1^{\mathcal{A}} = 0$.

[7] Showing that $\mathsf{test\text{-}real}_0$ and $\mathsf{test\text{-}real}_1$ as well as $\mathsf{test\text{-}td}_0$ and $\mathsf{test\text{-}td}_1$ are computationally indistinguishable again follows a reduction to simulation extractability.

the proof π. For this, we set $\chi = (\mathsf{pp}, \mathsf{pk}, \mathsf{ptd}, c)$ as $(\omega, \mathsf{st}_{\mathsf{Sim}}) \xleftarrow{\$} \mathsf{Sim.Ext}(\chi, \pi, \mathsf{st}_{\mathsf{Sim}})$. We let the oracle output $\perp$, if $(\chi, \omega) \notin \mathsf{R}$.

Let $\mathsf{st}_{\mathsf{Sim}}^{(0)}$ be the state of the simulator once the adversary outputs a decoder D'. During the experiment $\mathsf{test\text{-}real}$ and $\mathsf{test\text{-}td}$, we replace the proofs π generated for the decoder D' with simulated proofs. Crucially, each experiment starts with the simulator state $\mathsf{st}_{\mathsf{Sim}}^{(0)}$ (this will become important later in Claim 7.8). In the experiment $\mathsf{test\text{-}td}'$, we keep generating real proofs.

Claim 7.7 If PS is simulation-extractable, then for all PPT adversaries $\mathcal{A}$ the following inequalities hold

$$\begin{aligned}
|\Pr[\mathsf{GoodTr}_1] - \Pr[\mathsf{GoodTr}_0]| &\leq \mathsf{negl}(\lambda) \\
|\Pr[\mathsf{BadTr}_1] - \Pr[\mathsf{BadTr}_0]| &\leq \mathsf{negl}(\lambda) \\
|\Pr[\mathsf{test\text{-}real}_1] - \Pr[\mathsf{test\text{-}real}_0]| &\leq \mathsf{negl}(\lambda) \\
|\Pr[\mathsf{test\text{-}td}_1] - \Pr[\mathsf{test\text{-}td}_0]| &\leq \mathsf{negl}(\lambda) \\
\left|\Pr\left[\mathsf{test\text{-}td}'_1\right] - \Pr\left[\mathsf{test\text{-}td}'_0\right]\right| &\leq \mathsf{negl}(\lambda)
\end{aligned}$$

Proof. Let $\mathcal{A}$ be any PPT adversary. Consider a reduction $\mathcal{B}$ against the SimExt game of PS_{R}, and denote with b the internal bit of the SimExt game. If $b = 0$, $\mathcal{B}$ is playing against the real protocol, and if $b = 1$, $\mathcal{B}$ is playing against the simulator. Initially, $\mathcal{B}$ receives $\mathsf{pp}_{\mathsf{PS}_{\mathsf{R}}}$ and embeds it into $\Pi'.\mathsf{Setup}$. The reduction $\mathcal{B}$ simulates either game perfectly to $\mathcal{A}$, in particular, it simulates the oracle $\mathcal{O}^{\mathsf{PDec}}$ as follows: On input $(i, (c, \pi))$, instead of verifying the proof, query the $\mathcal{O}^{\mathsf{verify}}((\mathsf{pp}, \mathsf{pk}, \mathsf{ptd}, c), \pi)$ oracle of the challenger. If this oracle returns $\perp$, also return $\perp$. Otherwise, compute $d_i \leftarrow \Pi.\mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)$ and output d_i, which is possible since all secret keys are known to $\mathcal{B}$. Note that $\mathcal{B}$ simulates $\mathcal{O}_b^{\mathsf{PDec}}$ perfectly, where b is the internal bit of its challenger.

Once $\mathcal{A}$ outputs the decoder D', $\mathcal{B}$ can compute the quantities GoodTr, BadTr, $\mathsf{test\text{-}real}$, $\mathsf{test\text{-}td}$, and $\mathsf{test\text{-}td}'$. Note that to compute $\mathsf{test\text{-}real}$ and $\mathsf{test\text{-}td}$, $\mathcal{B}$ uses the $\mathcal{O}^{\mathsf{prove}}(\chi, \omega)$ oracle to attach proofs to the decoder's ciphertexts.

Let $E \in \{\mathsf{GoodTr}, \mathsf{BadTr}, \mathsf{test\text{-}real}, \mathsf{test\text{-}td}, \mathsf{test\text{-}td}'\}$. Observe that, if $b = 0$, $\mathcal{B}$ simulates Game_0 perfectly, and thus computes quantitiy E_0. If $b = 1$, $\mathcal{B}$ simulates Game_1 perfectly, and thus computes quantity E_1. Suppose that $\mathcal{B}$ outputs E as computed. Then, we get that

$$|\Pr[E_1] - \Pr[E_0]| \leq 2 \cdot \left|\Pr\left[\mathsf{SimExt}_{\mathcal{B}}^{\mathsf{PS}_{\mathsf{R}}, \mathsf{Sim}}(1^\lambda = 1)\right] - \frac{1}{2}\right|,$$

for all $E \in \{\mathsf{GoodTr}, \mathsf{BadTr}, \mathsf{test\text{-}real}, \mathsf{test\text{-}td}, \mathsf{test\text{-}td}'\}$, which is negligible by assumption. ∎

Observe that in Game_1, the following claim holds:

Claim 7.8 For all PPT adversaries $\mathcal{A}$ running in Game_1 it holds that

$$\Pr[\mathsf{test\text{-}real}_1] \leq \Pr[\mathsf{test\text{-}td}_1] + \mathsf{negl}(\lambda).$$

Proof. The main challenge in proving the claim is that the experiments $\mathsf{test\text{-}td}_1$ and $\mathsf{test\text{-}real}_1$ are not naturally independent across repetitions. In particular, both experiments carry over the internal simulator state $\mathsf{st}_{\mathsf{Sim}}$ between successive iterations. Thus, even though the individual repetitions are identically distributed, the success events in different repetitions may still be correlated by the evolving simulator state. As a result, we cannot apply a standard concentration bound, such as Hoeffding's inequality, directly.

Fortunately, though, our goal is only to bound the difference of the marginal probabilities $\Pr[\mathsf{test\text{-}real}_1] - \Pr[\mathsf{test\text{-}td}_1] \leq \mathsf{negl}(\lambda)$, and not to show that $\Pr[\mathsf{test\text{-}real}_1 = 1 \wedge \mathsf{test\text{-}td}_1 = 0] \leq \mathsf{negl}(\lambda)$.

Instead of analyzing two independent runs of the experiments, this allows us to analyze a coupling of the two experiments that preserves the distribution of both $\mathsf{test\text{-}real}_1$ and $\mathsf{test\text{-}td}_1$ *individually*. In particular, we next define such a coupling by fixing the challenge transcript, including the generated proofs and thus the simulator state.

Let $\mathsf{st}_{\mathsf{Sim}}^{(0)}$ be the state of the simulator immediately after the adversary outputs the decoder D'. Then, sample for every $i \in [Q]$

$$(k_0^{(i)}, c^{(i)}) \xleftarrow{\$} \Pi.\mathsf{Enc}(\mathsf{pk}); k_1^{(i)} \xleftarrow{\$} \mathcal{K}; b^{(i)} \xleftarrow{\$} \{0, 1\}$$

as well as $(\pi^{(i)}, \mathsf{st}_{\mathsf{Sim}}^{(i)}) \xleftarrow{\$} \mathsf{Sim.Sim}((\mathsf{pp}, \mathsf{pk}, \mathsf{ptd}, c^{(i)}), \mathsf{st}_{\mathsf{Sim}}^{(i-1)})$, and compute partial decryptions

$$\{d_u^{(i)} \xleftarrow{\$} \Pi'.\mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_u, (c^{(i)}, \pi^{(i)})\}_{u \in [n]}.$$

We denote the resulting transcript for the experiments as

$$T = \left(k_0^{(i)}, k_1^{(i)}, c^{(i)}, b^{(i)}, \pi^{(i)}, \{d_u^{(i)}\}_{u\in[n]}\right)_{i\in[Q]}.$$

Now, suppose that we run both tests on the same transcript T, but use fresh independent randomness for each decoder invocation. Let X_i be the indicator that the decoder successfully guesses $b^{(i)}$ when run in $\mathsf{test\text{-}real}_1$ using values from T_i, and Y_i be analogous for $\mathsf{test\text{-}td}_1$. This coupling clearly preserves the marginal distributions: When viewed on its own, the X-side sampled exactly the transcript and decoder randomness of $\mathsf{test\text{-}real}_1$, while the Y-side samples the exact transcript and decoder randomness of $\mathsf{test\text{-}td}_1$. Let $X = \sum_{i=1}^{Q} X_i$ and $Y = \sum_{i=1}^{Q} Y_i$. Then, we get that

$$\Pr[\mathsf{test\text{-}real}_1] = \Pr[X \geq Q \cdot (1/2 + 7\varepsilon/8)]$$

as well as

$$\Pr[\mathsf{test\text{-}td}_1] = \Pr[Y \geq Q \cdot (1/2 + 6\varepsilon/8)].$$

However, the only remaining randomness in the variables X_i and Y_i after conditioning on T is the fresh, independent decoder randomness. Hence, conditioned on T, we get that $X_i \equiv Y_i$ for all $i \in Q$.

Let $Z_i = X_i - Y_i \in [-1, 1]$ and $Z = \sum_{i=1}^{Q} Z_i$. Conditioned on T, we get

$$\mathbb{E}[Z_i \mid T] = \mathbb{E}[X_i - Y_i \mid T] = 0,$$

and all Z_i are independent. Hence, we can apply Hoeffding's inequality for every fixed transcript T.

$$\Pr\left[Z \geq \frac{Q\varepsilon}{8} \mid T\right] \leq \exp\left(-\frac{Q\varepsilon^2}{128}\right)$$

Since the above holds for every transcript, we can average over T such that

$$\Pr\left[Z \geq \frac{Q\varepsilon}{8}\right] = \mathbb{E}_T\left[\Pr\left[Z \geq \frac{Q\varepsilon}{8} \mid T\right]\right] \leq \exp\left(-\frac{Q\varepsilon^2}{128}\right),$$

which is negligible for $Q \geq 128\lambda/\varepsilon^2$.

Finally, under the coupling outlined above, we have

$$\Pr[\mathsf{test\text{-}real}_1] - \Pr[\mathsf{test\text{-}td}_1] = \mathbb{E}[\mathsf{test\text{-}real}_1 - \mathsf{test\text{-}td}_1] \leq \Pr[\mathsf{test\text{-}real}_1 = 1 \wedge \mathsf{test\text{-}td}_1 = 0],$$

and if $\mathsf{test\text{-}real}_1 = 1$, but $\mathsf{test\text{-}td}_1 = 0$ then by definition it must hold that

$$X \geq Q \cdot \left(\frac{1}{2} + \frac{7\varepsilon}{8}\right) \text{ and } Y < Q \cdot \left(\frac{1}{2} + \frac{6\varepsilon}{8}\right),$$

and thus $Z > Q\varepsilon/8$. As a result, we get that

$$\Pr[\mathsf{test\text{-}real}_1] - \Pr[\mathsf{test\text{-}td}_1] \leq \Pr\left[Z \geq \frac{Q\varepsilon}{8}\right] \leq \mathsf{negl}(\lambda),$$

for our choice of Q. ∎

$\underline{\mathsf{Game}_2}$. Next, we argue that the extracted witness ω in $\mathcal{O}^{\mathsf{PDec}}$ cannot be a solution to the hard relation H. For this, we add an additional abort condition to $\mathcal{O}^{\mathsf{PDec}}$, returning $\perp$ if $(\mathsf{ptd}, \omega) \in \mathsf{H}$ for the extracted witness ω.

Claim 7.9 If H is a hard relation, then for all PPT adversaries $\mathcal{A}$ it holds that

$$\begin{aligned} |\Pr[\mathsf{GoodTr}_2] - \Pr[\mathsf{GoodTr}_1]| &\leq \mathsf{negl}(\lambda) \\ |\Pr[\mathsf{BadTr}_2] - \Pr[\mathsf{BadTr}_1]| &\leq \mathsf{negl}(\lambda) \\ |\Pr[\mathsf{test\text{-}td}'_2] - \Pr[\mathsf{test\text{-}td}'_1]| &\leq \mathsf{negl}(\lambda) \end{aligned}$$

Proof. Let $\mathcal{A}$ be a distinguisher, and let $\mathcal{B}$ be a PPT reduction against the hardness of H. $\mathcal{B}$ initially receives Y from its challenger and sets $\mathsf{ptd} \leftarrow Y$ during Setup. It simulates either game to $\mathcal{A}$. When $\mathcal{A}$ makes an $\mathcal{O}^{\mathsf{PDec}}(i, (c, \pi))$ query that hits the additional abort condition, i.e. $(\mathsf{ptd}, \omega) \in \mathsf{H}$ for the extracted witness ω, then $\mathcal{B}$ outputs ω. If no such query occurs, $\mathcal{B}$ outputs $\perp$.

Note that the events GoodTr, BadTr, and $\mathsf{test\text{-}td}'$ are *identical* between Game_1, and Game_2, unless $\mathcal{A}$ triggers the additional abort condition. Hence, their differences can be directly bound by $\mathcal{B}$'s advantage. ∎

We conclude our proof with a reduction $\mathcal{B}$ from Game_2 to traceability of Π. Crucially, the reduction must simulate $\mathcal{O}^{\mathsf{PDec}}$ as defined under Game_2 (henceforth, $\mathcal{O}_2^{\mathsf{PDec}}$), but can only make use of a partial decryption oracle for valid ciphertexts $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$ offered by its challenger.

$\mathcal{B}$ simulates $\Pi.\mathsf{Setup}$ by forwarding messages between its challenger and $\mathcal{A}$.

Simulating $\mathcal{O}^{\mathsf{corrupt}}$. Whenever $\mathcal{A}$ issues an $\mathcal{O}^{\mathsf{corrupt}}$ query for some party i, $\mathcal{B}$ forwards that query to its challenger's $\mathcal{O}^{\mathsf{corrupt}}$ oracle, and forwards the secret key back to $\mathcal{A}$. As a result, $\mathcal{B}$ corrupts the same parties as traitors as $\mathcal{A}$ ($\mathcal{T}^{\mathcal{A}} = \mathcal{T}^{\mathcal{B}}$).

Simulating $\mathcal{O}_2^{\mathsf{PDec}}$. Upon close inspection of $\mathcal{O}_2^{\mathsf{PDec}}(i,(c,\pi))$, observe that $\mathcal{B}$ must return $\perp$ whenever any of the following conditions is met:

- $\mathsf{Verify}((\mathsf{pp},\mathsf{pk},\mathsf{ptd},c),\pi) = 0$
- $((\mathsf{pp},\mathsf{pk},\mathsf{ptd},c),\omega) \notin \mathsf{R}$
- $(\mathsf{ptd},\omega) \in \mathsf{H}$

Note that the second and third conditions together imply that $\perp$ is returned unless $\Pi.\mathsf{Enc}(\mathsf{pk};\omega) = c$. In this case, however, $\mathcal{B}$ can simply query its own challenger for $\mathcal{O}^{\mathsf{PDec\text{-}Val}}(i, r_{\mathsf{Enc}} := \omega)$ to obtain a partial decryption d_i on c without knowing sk_i, and thus simulate $\mathcal{O}_2^{\mathsf{PDec}}$ perfectly.

When $\mathcal{A}$ outputs a decoder D', then $\mathcal{B}$ constructs the decoder D as mandated by Π'. I.e., on input (k,c), the decoder D computes $\pi \xleftarrow{\$} \mathsf{PS.Prove}((\mathsf{pp},\mathsf{pk},\mathsf{ptd},c),\mathsf{td})$, and forwards $(k,(c,\pi))$ to D'. Once D' outputs $(\mathcal{I}_{\mathsf{pd}},\mathsf{st})$, D outputs the same. When given $(\mathsf{st},\{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}})$, D runs $D'(\mathsf{st},\mathcal{I}_{\mathsf{pd}})$ and returns the bit output by D'.

Finally, $\mathcal{B}$ outputs the decoder D to its challenger.

Analysis. We start with the BadTr case. Clearly, whenever $\mathsf{BadTr}^{\mathcal{B}}$ happens, then also $\mathsf{BadTr}_2^{\mathcal{A}}$ happens, since $\mathcal{B}$'s simulation is perfect, and $\mathcal{T}^{\mathcal{A}} = \mathcal{T}^{\mathcal{B}}$. We get that

$$\Pr\left[\mathsf{BadTr}_2^{\mathcal{A}}\right] = \Pr\left[\mathsf{BadTr}^{\mathcal{B}}\right],$$

which is negligible by the traceability of Π under $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$.

Moving on to the GoodTr case, observe that, for the same reason, if $\mathsf{GoodTr}^{\mathcal{B}}$ happens, then so does $\mathsf{GoodTr}_2^{\mathcal{A}}$.

Finally, we want to show that if $\mathsf{test\text{-}td}_2'$ happens, then $\mathsf{GoodDec}^{\mathcal{B}}$ also happens, except with negligible probability. To see that this is the case, observe that the experiment underpinning $\mathsf{test\text{-}td}_2'$ is essentially an approximation of $\mathsf{GoodDec}^{\mathcal{B}}$, as defined by the function $P_{\Pi}^{\mathcal{B}}(D)$:

$$\begin{aligned} P_{\Pi}^{\mathcal{B}}(D) &= \Pr\left[D(\mathsf{st},\{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}}) = b \colon \begin{array}{c} (k_0,c) \xleftarrow{\$} \Pi.\mathsf{Enc}(\mathsf{pk}); k_1 \xleftarrow{\$} \mathcal{K}; b \xleftarrow{\$} \{0,1\} \\ (\mathcal{I}_{\mathsf{pd}},\mathsf{st}) \xleftarrow{\$} D(k_b,c) \\ \forall i \in \mathcal{I}_{\mathsf{pd}} \colon d_i \leftarrow \Pi.\mathsf{PDec}(\mathsf{pk},\mathsf{sk}_i,c) \end{array} \right] \\ &= \Pr\left[D'(\mathsf{st},\{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}}) = b \colon \begin{array}{c} (k_0,c) \xleftarrow{\$} \Pi.\mathsf{Enc}(\mathsf{pk}); k_1 \xleftarrow{\$} \mathcal{K}; b \xleftarrow{\$} \{0,1\} \\ \pi \xleftarrow{\$} \mathsf{PS.Prove}((\mathsf{pp},\mathsf{pk},\mathsf{ptd},c),\mathsf{td}) \\ (\mathcal{I}_{\mathsf{pd}},\mathsf{st}) \xleftarrow{\$} D'(k_b,(c,\pi)) \\ \forall i \in \mathcal{I}_{\mathsf{pd}} \colon d_i \leftarrow \Pi.\mathsf{PDec}(\mathsf{pk},\mathsf{sk}_i,c) \end{array} \right] =: \gamma \end{aligned}$$

Now, γ essentially is the probability that the decoder succeeds an a given single repetition of the $\mathsf{test\text{-}td}_2'$ experiment, and thus, we expect to observe $Q\cdot\gamma = Q\cdot P_{\Pi}^{\mathcal{B}}(D)$ successful repetitions. Also, recall that $\mathsf{test\text{-}td}_2' = 1$, if at least a fraction $1/2 + 5\varepsilon/8$ of the Q repetitions are successful.

As a result, if the decoder D' is successful in at least a fraction $1/2 + 5\varepsilon/8$ repetitions inside the experiment $\mathsf{test\text{-}td}_2'$, then $\mathcal{B}$ must produce a decoder D that distinguishes successfully with probability $P_{\Pi}^{\mathcal{B}}(D) \geq 1/2 + \varepsilon/2$, except with negligible probability. A simple Hoeffding bound formalizes the above intuition, and we get that

$$\Pr\left[\mathsf{test\text{-}td}_2'\right] \leq \Pr\left[\mathsf{GoodDec}^{\mathcal{B}}\right] + \exp(-Q\varepsilon^2/32),$$

which is negligible if $Q \geq 32\lambda/\varepsilon^2$.

By the traceability of Π under $\mathcal{O}^{\mathsf{PDec\text{-}Val}}$, we get that

$$\Pr\left[\mathsf{GoodTr}_2^{\mathcal{A}}\right] = \Pr\left[\mathsf{GoodTr}^{\mathcal{B}}\right] \geq \Pr\left[\mathsf{GoodDec}^{\mathcal{B}}\right] - \mathsf{negl}(\lambda) \geq \Pr\left[\mathsf{test\text{-}td}_2'\right] - \mathsf{negl}'(\lambda) \tag{1}$$

for appropriate negligible functions negl and negl'.

Putting it all together. Finally, we can apply Claims 7.6 7.7, 7.8, and 7.9 to get that for all PPT adversaries $\mathcal{A}$, it holds that

$$\Pr\left[\mathsf{BadTr}^{\mathcal{A}}\right] \leq \Pr\left[\mathsf{BadTr}_2^{\mathcal{A}}\right] + \mathsf{negl}(\lambda) \leq \mathsf{negl}'(\lambda)$$

and

$$\begin{aligned}
\Pr\left[\mathsf{GoodTr}^{\mathcal{A}}\right] &\geq \Pr\left[\mathsf{GoodTr}_2^{\mathcal{A}}\right] - \mathsf{negl}(\lambda) \\
&\overset{\text{Eq. 1}}{\geq} \Pr\left[\mathsf{test\text{-}td}_2'\right] - \mathsf{negl}'(\lambda) \\
&\overset{\text{Claims 7.7, 7.9}}{\geq} \Pr\left[\mathsf{test\text{-}td}_0'\right] - \mathsf{negl}''(\lambda) \\
&\overset{\text{Claim 7.6}}{\geq} \Pr[\mathsf{test\text{-}td}_0] - \mathsf{negl}'''(\lambda) \\
&\overset{\text{Claim 7.7}}{\geq} \Pr[\mathsf{test\text{-}td}_1] - \mathsf{negl}^{(4)}(\lambda) \\
&\overset{\text{Claim 7.8}}{\geq} \Pr[\mathsf{test\text{-}real}_1] - \mathsf{negl}^{(5)}(\lambda) \\
&\overset{\text{Claim 7.7}}{\geq} \Pr[\mathsf{test\text{-}real}_1] - \mathsf{negl}^{(6)}(\lambda) \\
&\overset{\text{Claim 7.6}}{\geq} \Pr\left[\mathsf{GoodDec}^{\mathcal{A}}\right] - \mathsf{negl}^{(7)}(\lambda),
\end{aligned}$$

for appropriate negligible functions. ■

Note that SIND-CPA security is trivially maintained, assuming that PS_{R} is zero-knowledge: The proof works simply by replacing the NIZK-PoK π attached to the challenge ciphertext with a simulated one, and then reducing to SIND-CPA security of Π.

References

[1] Agarwal, A., Fernando, R., Pinkas, B.: Efficiently-thresholdizable batched identity based encryption, with applications. In: Kalai, Y.T., Kamara, S.F. (eds.) CRYPTO 2025, Part III. LNCS, vol. 16002, pp. 69–100. Springer, Cham (Aug 2025). https://doi.org/10.1007/978-3-032-01881-6_3 (Cited on page 1.)

[2] Ballard, L., Green, M., de Medeiros, B., Monrose, F.: Correlation-resistant storage via keyword-searchable encryption. Cryptology ePrint Archive, Report 2005/417 (2005), https://eprint.iacr.org/2005/417 (Cited on page 9, 10.)

[3] Bebel, J., Ojha, D.: Ferveo: Threshold decryption for mempool privacy in BFT networks. Cryptology ePrint Archive, Report 2022/898 (2022), https://eprint.iacr.org/2022/898 (Cited on page 1.)

[4] Beimel, A.: Secret-sharing schemes: A survey. In: International conference on coding and cryptology. pp. 11–46. Springer (2011) (Cited on page 58.)

[5] Bhattacharyya, R., Bormet, J., Faust, S., Mukherjee, P., Othman, H.: CCA-secure traceable threshold (ID-based) encryption and application. In: Huang, C.Y., Chen, J.C., Shieh, S.P., Lie, D., Cortier, V. (eds.) ACM CCS 2025. pp. 2324–2338. ACM Press (Oct 2025). https://doi.org/10.1145/3719027.3744876 (Cited on page 1, 3.)

[6] Boneh, D., Boyen, X., Goh, E.J.: Hierarchical identity based encryption with constant size ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Berlin, Heidelberg (May 2005). https://doi.org/10.1007/11426639_26 (Cited on page 10, 56.)

[7] Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Berlin, Heidelberg (Aug 2004). https://doi.org/10.1007/978-3-540-28628-8_3 (Cited on page 9, 10.)

[8] Boneh, D., Bünz, B., Nayak, K., Rotem, L., Shoup, V.: Context-dependent threshold decryption and its applications. In: Hanaoka, G., Yang, B.Y. (eds.) ASIACRYPT 2025, Part VI. LNCS, vol. 16250, pp. 506–538. Springer, Singapore (Dec 2025). https://doi.org/10.1007/978-981-95-5119-4_16 (Cited on page 2.)

[9] Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Berlin, Heidelberg (Aug 2001). https://doi.org/10.1007/3-540-44647-8_13 (Cited on page 10.)

[10] Boneh, D., Laufer, E., Tas, E.N.: Batch decryption without epochs and its application to encrypted mempools. Cryptology ePrint Archive, Report 2025/1254 (2025), https://eprint.iacr.org/2025/1254 (Cited on page 1.)

[11] Boneh, D., Partap, A., Rotem, L.: Accountability for misbehavior in threshold decryption via threshold traitor tracing. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part VII. LNCS, vol. 14926, pp. 317–351. Springer, Cham (Aug 2024). https://doi.org/10.1007/978-3-031-68394-7_11 (Cited on page 1, 3, 4.)

[12] Boneh, D., Partap, A., Rotem, L.: Traceable secret sharing: Strong security and efficient constructions. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part V. LNCS, vol. 14924, pp. 221–256. Springer, Cham (Aug 2024). https://doi.org/10.1007/978-3-031-68388-6_9 (Cited on page 4.)

[13] Boneh, D., Partap, A., Rotem, L.: Traceable verifiable random functions. In: Kalai, Y.T., Kamara, S.F. (eds.) CRYPTO 2025, Part II. LNCS, vol. 16001, pp. 203–234. Springer, Cham (Aug 2025). https://doi.org/10.1007/978-3-032-01878-6_7 (Cited on page 4.)

[14] Boneh, D., Waters, B.: A fully collusion resistant broadcast, trace, and revoke system. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM CCS 2006. pp. 211–220. ACM Press (Oct / Nov 2006). https://doi.org/10.1145/1180405.1180432 (Cited on page 2.)

[15] Boneh, D., Zhandry, M.: Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 480–499. Springer, Berlin, Heidelberg (Aug 2014). https://doi.org/10.1007/978-3-662-44371-2_27 (Cited on page 4.)

[16] Bormet, J., Choudhuri, A.R., Faust, S., Garg, S., Othman, H., Policharla, G.V., Qu, Z., Wang, M.: BEAST-MEV: Batched threshold encryption with silent setup for MEV prevention. Cryptology ePrint Archive, Report 2025/1419 (2025), https://eprint.iacr.org/2025/1419 (Cited on page 1.)

[17] Bormet, J., Dziembowski, S., Faust, S., Lizurej, T., Mielniczuk, M.: Strong secret sharing with snitching. In: Kalai, Y.T., Kamara, S.F. (eds.) CRYPTO 2025, Part IV. LNCS, vol. 16003, pp. 295–327. Springer, Cham (Aug 2025). https://doi.org/10.1007/978-3-032-01884-7_10 (Cited on page 4.)

[18] Bormet, J., Faust, S., Othman, H.: Threshold traitor tracing revisited: Insider attacks and multi-traitor tracing. In: TCC 2026 (2026), https://eprint.iacr.org/2026/422 (Cited on page 1, 2, 3, 4, 5, 10, 11, 56, 58.)

[19] Bormet, J., Faust, S., Othman, H., Qu, Z.: BEAT-MEV: Epochless approach to batched threshold encryption for MEV prevention. In: Bauer, L., Pellegrino, G. (eds.) USENIX Security 2025. pp. 3457–3476. USENIX Association (Aug 2025), https://www.usenix.org/conference/usenixsecurity25/presentation/bormet (Cited on page 1.)

[20] Bormet, J., Hofmann, J., Othman, H.: Traceable threshold encryption without a trusted dealer. In: Hanaoka, G., Yang, B.Y. (eds.) ASIACRYPT 2025, Part VI. LNCS, vol. 16250, pp. 471–505. Springer, Singapore (Dec 2025). https://doi.org/10.1007/978-981-95-5119-4_15 (Cited on page 1, 2, 3, 6, 9, 13.)

[21] Boyen, X.: The uber-assumption family (invited talk). In: Galbraith, S.D., Paterson, K.G. (eds.) PAIRING 2008. LNCS, vol. 5209, pp. 39–56. Springer, Berlin, Heidelberg (Sep 2008). https://doi.org/10.1007/978-3-540-85538-5_3 (Cited on page 56.)

[22] Brzuska, C., Klooß, M., Woo, I.K.Y.: Threshold public-key encryption: Definitions, relations, and CPA-to-CCA transforms. In: Bai, S., Persichetti, E. (eds.) PKC 2026, Part IV. LNCS, vol. 16554, pp. 162–197. Springer, Cham (May 2026). https://doi.org/10.1007/978-3-032-26740-5_6 (Cited on page 26.)

[23] Canard, S., Papon, N., Phan, D.H.: Public traceability in threshold decryption. CiC **2**(2), 21 (2025). https://doi.org/10.62056/akjb0lmol (Cited on page 1, 3.)

[24] Chakrabarti, A., Maitra, M., Mondal, A., Sehgal, K.: Silent threshold traitor tracing & enhancing mempool privacy. In: Huang, C.Y., Chen, J.C., Shieh, S.P., Lie, D., Cortier, V. (eds.) ACM CCS 2025. pp. 1769–1783. ACM Press (Oct 2025). https://doi.org/10.1145/3719027.3765099 (Cited on page 1, 3.)

[25] Chen, M., Chiesa, A., Gur, T., O'Connor, J., Spooner, N.: Proof-carrying data from arithmetized random oracles. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part II. LNCS, vol. 14005, pp. 379–404. Springer, Cham (Apr 2023). https://doi.org/10.1007/978-3-031-30617-4_13 (Cited on page 28.)

[26] Chen, M., Chiesa, A., Spooner, N.: On succinct non-interactive arguments in relativized worlds. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 336–366. Springer, Cham (May / Jun 2022). https://doi.org/10.1007/978-3-031-07085-3_12 (Cited on page 28.)

[27] Chen, Y., Vaikuntanathan, V., Waters, B., Wee, H., Wichs, D.: Traitor-tracing from LWE made simple and attribute-based. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, Part II. LNCS, vol. 11240, pp. 341–369. Springer, Cham (Nov 2018). https://doi.org/10.1007/978-3-030-03810-6_13 (Cited on page 4.)

[28] Chiesa, A., Ojha, D., Spooner, N.: Fractal: Post-quantum and transparent recursive proofs from holography. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part I. LNCS, vol. 12105, pp. 769–793. Springer, Cham (May 2020). https://doi.org/10.1007/978-3-030-45721-1_27 (Cited on page 28.)

[29] Chor, B., Fiat, A., Naor, M.: Tracing traitors. In: Desmedt, Y. (ed.) CRYPTO'94. LNCS, vol. 839, pp. 257–270. Springer, Berlin, Heidelberg (Aug 1994). https://doi.org/10.1007/3-540-48658-5_25 (Cited on page 4.)

[30] Choudhuri, A.R., Garg, S., Piet, J., Policharla, G.V.: Mempool privacy via batched threshold encryption: Attacks and defenses. In: Balzarotti, D., Xu, W. (eds.) USENIX Security 2024. USENIX Association (Aug 2024), https://www.usenix.org/conference/usenixsecurity24/presentation/choudhuri (Cited on page 1.)

[31] Das, S., Datta, P., Partap, A., Sasmal, S., Zhandry, M.: Optimal threshold traitor tracing. In: Daemen, J., Thomé, E. (eds.) EUROCRYPT 2026, Part V. LNCS, vol. 16545, pp. 486–516. Springer, Cham (May 2026). https://doi.org/10.1007/978-3-032-25330-9_17 (Cited on page 1, 2, 3, 4, 11.)

[32] Desmedt, Y.: Society and group oriented cryptography: A new concept. In: Pomerance, C. (ed.) CRYPTO'87. LNCS, vol. 293, pp. 120–127. Springer, Berlin, Heidelberg (Aug 1988). https://doi.org/10.1007/3-540-48184-2_8 (Cited on page 1.)

[33] Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO'89. LNCS, vol. 435, pp. 307–315. Springer, New York (Aug 1990). https://doi.org/10.1007/0-387-34805-0_28 (Cited on page 1.)

[34] Dziembowski, S., Faust, S., Lizurej, T., Mielniczuk, M.: Secret sharing with snitching. In: Luo, B., Liao, X., Xu, J., Kirda, E., Lie, D. (eds.) ACM CCS 2024. pp. 840–853. ACM Press (Oct 2024). https://doi.org/10.1145/3658644.3690296 (Cited on page 4.)

[35] Dziembowski, S., Faust, S., Luhn, J.: Shutter network: Private transactions from threshold cryptography. Cryptology ePrint Archive, Report 2024/1981 (2024), https://eprint.iacr.org/2024/1981 (Cited on page 1.)

[36] Farràs, O., Guiot, M.: Traceable secret sharing schemes for general access structures. In: Daemen, J., Thomé, E. (eds.) EUROCRYPT 2026, Part III. LNCS, vol. 16543, pp. 3–32. Springer, Cham (May 2026). https://doi.org/10.1007/978-3-032-25324-8_1 (Cited on page 4.)

[37] Fuchsbauer, G., Wolf, M.: Concurrently secure blind Schnorr signatures. In: Joye, M., Leander, G. (eds.) EUROCRYPT 2024, Part II. LNCS, vol. 14652, pp. 124–160. Springer, Cham (May 2024). https://doi.org/10.1007/978-3-031-58723-8_5 (Cited on page 28.)

[38] Gong, J., Luo, J., Wee, H.: Traitor tracing with $N^{1/3}$-size ciphertexts and $O(1)$-size keys from k-Lin. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part III. LNCS, vol. 14006, pp. 637–668. Springer, Cham (Apr 2023). https://doi.org/10.1007/978-3-031-30620-4_21 (Cited on page 4.)

[39] Goyal, R., Koppula, V., Waters, B.: Collusion resistant traitor tracing from learning with errors. In: Diakonikolas, I., Kempe, D., Henzinger, M. (eds.) 50th ACM STOC. pp. 660–670. ACM Press (Jun 2018). https://doi.org/10.1145/3188745.3188844 (Cited on page 4.)

[40] Goyal, R., Koppula, V., Waters, B.: New approaches to traitor tracing with embedded identities. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019, Part II. LNCS, vol. 11892, pp. 149–179. Springer, Cham (Dec 2019). https://doi.org/10.1007/978-3-030-36033-7_6 (Cited on page 4.)

[41] Goyal, R., Quach, W., Waters, B., Wichs, D.: Broadcast and trace with N^{ϵ} ciphertext size from standard assumptions. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 826–855. Springer, Cham (Aug 2019). https://doi.org/10.1007/978-3-030-26954-8_27 (Cited on page 4.)

[42] Goyal, R., Vusirikala, S., Waters, B.: Collusion resistant broadcast and trace from positional witness encryption. In: Lin, D., Sako, K. (eds.) PKC 2019, Part II. LNCS, vol. 11443, pp. 3–33. Springer, Cham (Apr 2019). https://doi.org/10.1007/978-3-030-17259-6_1 (Cited on page 4.)

[43] Goyal, V., Jain, A., Partap, A.: Traceable secret sharing revisited. In: Daemen, J., Thomé, E. (eds.) EUROCRYPT 2026, Part III. LNCS, vol. 16543, pp. 33–63. Springer, Cham (May 2026). https://doi.org/10.1007/978-3-032-25324-8_2 (Cited on page 4.)

[44] Goyal, V., Song, Y., Srinivasan, A.: Traceable secret sharing and applications. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part III. LNCS, vol. 12827, pp. 718–747. Springer, Cham, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84252-9_24 (Cited on page 4.)

[45] Joux, A.: A one round protocol for tripartite diffie–hellman. In: Bosma, W. (ed.) Algorithmic Number Theory. pp. 385–393. Springer Berlin Heidelberg, Berlin, Heidelberg (2000) (Cited on page 10.)

[46] Maurer, U.M.: Abstract models of computation in cryptography (invited paper). In: Smart, N.P. (ed.) 10th IMA International Conference on Cryptography and Coding. LNCS, vol. 3796, pp. 1–12. Springer, Berlin, Heidelberg (Dec 2005). https://doi.org/10.1007/11586821_1 (Cited on page 56.)

[47] Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) EUROCRYPT'97. LNCS, vol. 1233, pp. 256–266. Springer, Berlin, Heidelberg (May 1997). https://doi.org/10.1007/3-540-69053-0_18 (Cited on page 10, 56.)

[48] Waters, B., Wu, D.J.: Silent threshold cryptography from pairings: Expressive policies in the plain model. In: Daemen, J., Thomé, E. (eds.) EUROCRYPT 2026, Part V. LNCS, vol. 16545, pp. 517–547. Springer, Cham (May 2026). https://doi.org/10.1007/978-3-032-25330-9_18 (Cited on page 7, 8, 10, 23.)

[49] Zhandry, M.: Optimal traitor tracing from pairings. In: Fehr, S., Fouque, P.A. (eds.) EUROCRYPT 2025, Part III. LNCS, vol. 15603, pp. 305–335. Springer, Cham (May 2025). https://doi.org/10.1007/978-3-031-91131-6_11 (Cited on page 4.)

Appendices

A Additional Preliminaries

Definition A.1 (Commitment Schemes). A commitment scheme is a tuple of algorithms $\Pi_{\mathsf{Com}} = (\mathsf{Setup}, \mathsf{Com})$ with the following syntax

- $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$. Given the security parameter λ, output public parameters pp.
- $\mathsf{com} \leftarrow \mathsf{Com}(\mathsf{pp}, m; r)$. Given pp, a message m and uniformly chosen randomness r, output a commitment com.

Game $\mathsf{Hiding}_{\mathcal{A}}(1^\lambda)$	Game $\mathsf{Binding}_{\mathcal{A}}(1^\lambda)$
$\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$	$\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
$((m_0, m_1), \mathsf{st}) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pp}) \quad // \ \lvert m_0\rvert = \lvert m_1\rvert$	$(\mathsf{com}, (m_0, r_0), (m_1, r_1)) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pp})$
$b \xleftarrow{\$} \{0,1\}$	**if** $m_0 \neq m_1 \wedge \mathsf{com} = \mathsf{Com}(m_0; r_0) \wedge \mathsf{com} = \mathsf{Com}(m_1; r_1)$ **then**
$\mathsf{com} \xleftarrow{\$} \mathsf{Com}(\mathsf{pp}, m_b)$	**return** 1
$b' \xleftarrow{\$} \mathcal{A}(\mathsf{st}, \mathsf{com})$	**return** 0
return $b \stackrel{?}{=} b'$	

Figure 13: The computational hiding and binding games for commitment schemes

Definition A.2 (Computational Hiding). We say that a commitment scheme Π_{Com} is computationally hiding if for all PPT adversaries $\mathcal{A}$, the following function is negligible in λ

$$\mathsf{Adv}^{\text{hiding}}_{\mathcal{A},\Pi_{\mathsf{Com}}}(1^\lambda) := \left|\Pr\left[\mathsf{Hiding}_{\mathcal{A}}(1^\lambda) = 1\right] - \frac{1}{2}\right|,$$

where the Hiding game is defined in Fig. 13.

Definition A.3 (Computational Binding). We say that a commitment scheme Π_{Com} is computationally binding if for all PPT adversaries $\mathcal{A}$, the following function is negligible in λ

$$\mathsf{Adv}^{\text{binding}}_{\mathcal{A},\Pi_{\mathsf{Com}}}(1^\lambda) := \Pr\left[\mathsf{Binding}\mathcal{A}(1^\lambda) = 1\right],$$

where the $\mathsf{Binding}$ game is defined in Fig. 13.

Theorem A.4 (Azuma–Hoeffding inequality). *Let $(X_i)_{i=0}^n$ be a real-valued martingale with respect to a filtration $(\mathcal{F}_i)_{i=0}^n$. Suppose there exist constants $c_1, \ldots, c_n \geq 0$ such that, for every $i \in [n]$,*

$$|X_i - X_{i-1}| \leq c_i.$$

Then, for every $t \geq 0$,

$$\Pr[X_n - X_0 \geq t] \leq \exp\left(-\frac{t^2}{2\sum_{i=1}^n c_i^2}\right).$$

Consequently,

$$\Pr[|X_n - X_0| \geq t] \leq 2\exp\left(-\frac{t^2}{2\sum_{i=1}^n c_i^2}\right).$$

The following lemma follows directly from Theorem A.4. We provide the proof for completeness.

Lemma A.5 *Let S_n denote the set of permutations over the integers $[n]$. Let $f : S_n \to \mathbb{R}$, and let (k, ℓ) denote the transposition that swaps the entries in positions k and ℓ. For a permutation $\pi = (\pi_1, \ldots, \pi_n) \in S_n$, we write $\pi \circ (k, \ell)$ for the permutation obtained from π by exchanging π_k and π_ℓ and leaving all other entries unchanged.*

Suppose that for every $\pi \in S_n$ and every pair $1 \leq k < \ell \leq n$,

$$\left|f(\pi) - f\left(\pi \circ (k, \ell)\right)\right| \leq c_k.$$

Let Y be uniformly distributed over S_n. Then, for every $\epsilon > 0$,

$$\Pr(|f(Y) - \mathbb{E}[f(Y)]| \geq \epsilon) \leq 2\exp\left(-\frac{\epsilon^2}{2\sum_{k=1}^n c_k^2}\right).$$

Proof. Let $(Y_1, \ldots, Y_n)$ be a uniformly random permutation of $[n]$, and consider the martingale

$$X_k = \mathbb{E}[f(Y_1, \ldots, Y_n) \mid Y_1, \ldots, Y_k], \qquad k = 0, \ldots, n.$$

Thus,

$$X_0 = \mathbb{E}[f(Y_1, \ldots, Y_n)] \qquad \text{and} \qquad X_n = f(Y_1, \ldots, Y_n).$$

We want to prove that, for every $k = 0, \ldots, n-1$,

$$|X_{k+1} - X_k| \leq c_{k+1}. \tag{2}$$

Fix $Y_1 = y_1, \ldots, Y_k = y_k$, and let A be the set of values that have not yet appeared among $y_1, \ldots, y_k$. For each possible value $a \in A$ of Y_{k+1}, we define

$$g(a) = \mathbb{E}[f(Y_1, \ldots, Y_n) \mid Y_1 = y_1, \ldots, Y_k = y_k, Y_{k+1} = a].$$

We first prove the following claim:

Claim A.6 For every $a, b \in A$ it holds that

$$|g(a) - g(b)| \leq c_{k+1}. \tag{3}$$

Proof. To see this, let

$$\Omega_a = \{\pi \in S_n : \pi_1 = y_1, \ldots, \pi_k = y_k, \pi_{k+1} = a\},$$

and define Ω_b analogously. Conditioned on $Y_1 = y_1, \ldots, Y_k = y_k, Y_{k+1} = a$, the unifom random permutation $(Y_1, \ldots, Y_n)$ is also uniformly distributed over Ω_a. Hence, as π is chosen uniformly from Ω_a, then

$$g(a) = \mathbb{E}[f(\pi)] = \frac{1}{|\Omega_a|} \sum_{\pi \in \Omega_a} f(\pi).$$

Similarly, as π' is uniformly distributed over Ω_b, then

$$g(b) = \mathbb{E}[f(\pi')] = \frac{1}{|\Omega_b|} \sum_{\pi' \in \Omega_b} f(\pi').$$

We now couple these two random permutations. Take $\pi \in \Omega_a$ for $a \in A$. Since $b \in A$ has not appeared among the first $k+1$ entries of π, it occurs at some position $\ell > k+1$. Let π' be obtained from π by swapping the entries in positions $k+1$ and ℓ. Then, $\pi' \in \Omega_b$. This operation is reversible, so it defines a bijection $\Omega_a \longrightarrow \Omega_b$. Therefore, if π is uniform over Ω_a, then the corresponding π' is uniform over Ω_b. Thus this bijection gives a coupling of the two conditional distributions. Under this coupling, π and π' differ by a single transposition involving position $k+1$. Therefore, by the hypothesis of the lemma,

$$|f(\pi) - f(\pi')| \leq c_{k+1}.$$

Thus, we obtain

$$\begin{aligned} |g(a) - g(b)| &= |\mathbb{E}[f(\pi)] - \mathbb{E}[f(\pi')]| \\ &= |\mathbb{E}[f(\pi) - f(\pi')]| \\ &\leq \mathbb{E}[|f(\pi) - f(\pi')|] \\ &\leq c_{k+1}. \end{aligned}$$

■

We now deduce Equation 2. Note first that, if $Y_{k+1} = a$, then $X_{k+1} = g(a)$. On the other hand, before revealing the value of Y_{k+1}, we average over all its possible values. Therefore,

$$X_k = \frac{1}{|A|} \sum_{a \in A} g(a).$$

In other words, X_k is the average of all the possible values that X_{k+1} can take after the next entry of the permutation is revealed. Claim A.6 holds for every $a, b \in A$, and thus in, in particular it holds for

$a = \max_{c \in A} g(c)$ and $b = \min_{c \in A} g(c)$ that, $|\max_{a \in A} g(a) - \min_{a \in A} g(a)| \leq c_{k+1}$. Since the average of a collection of numbers lies between its minimum and maximum,

$$\min_{a \in A} g(a) \leq X_k \leq \max_{a \in A} g(a).$$

Also, $X_{k+1} = g(Y_{k+1})$ is one of these values. Consequently,

$$|X_{k+1} - X_k| \leq |\max_{a \in A} g(a) - \min_{a \in A} g(a)| \leq c_{k+1}.$$

Therefore the martingale $X_0, \ldots, X_n$ satisfies the hypotheses of the Azuma–Hoeffding inequality (Theorem A.4), and hence

$$\Pr(|f(Y_1, \ldots, Y_n) - \mathbb{E}[f(Y_1, \ldots, Y_n)]| \geq \epsilon) \leq 2 \exp\left(-\frac{\epsilon^2}{2\sum_{k=1}^{n} c_k^2}\right),$$

which completes the proof. ∎

B Full TPLBE-PD Definition

We give an overview of the full definition of the algorithms making up TPLBE-PD:

- $(\mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{i \in [n]}, \mathsf{tk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, n, t)$. Setup is a (possibly) interactive, multi-round protocol between a trusted party $\mathbb{D}$, n committee members $\{\mathbb{P}_i\}_{i \in [n]}$, and a tracer $\mathbb{T}$. At the end of the protocol, each party $i \in [n]$ receives sk_i as private output, and a tracing key tk is private output for the tracer. The public parameters pp and the public key pk are public outputs. All subsequent algorithms receive pp as implicit input.
- $(k, c) \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk})$. Enc is a probabilistic algorithm that, given a public key pk, outputs a ciphertext c and an encapsulated key $k \in \mathcal{K}$.
- $(k, c) \xleftarrow{\$} \mathsf{TrEnc}(\mathsf{tk}, j)$. Given the tracing key tk and an index $j \in [0, n]$, the TrEnc algorithm outputs a ciphertext c and an encapsulated key k.
- $d_i \leftarrow \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)$. Given the public key pk, a secret key share sk_i on behalf of party $i \in [n]$, the PDec algorithm outputs a partial decryption d_i.
- $d_i \leftarrow \mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$. Given the public key pk, an index $i \in [n]$, a ciphertext c, and encryption randomness r_{Enc}, the SimPD algorithm outputs a simulated partial decryption d_i.
- $k \leftarrow \mathsf{Comb}(\mathsf{pk}, c, S, \{d_i\}_{i \in S})$. Given the public key pk, a ciphertext c, a set of partial decryptions $\{d_i\}_{i \in S}$ on behalf of parties in $S \subseteq [n]$, the Comb algorithm outputs the encapsulated key k.

We also state the formal correctness notion for TPLBE-PD schems.

Definition B.1 (Correctness). A TPLBE-PD is correct, if there exists a negligible function negl such that for all $\lambda \in \mathbb{N}$, all $n \in \mathbb{N}$, all t with $0 < t \leq n$, and all $S \subseteq [n]$ with $|S| \geq t$ it holds that

$$\Pr\left[\begin{array}{c}\mathsf{Comb}(\mathsf{pk}, c, S, \{d_i\}_S) \\ = k\end{array} : \begin{array}{c}(\mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{[n]}, \mathsf{tk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, n, t) \\ (k, c) \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}) \\ \forall i \in S\colon d_i \leftarrow \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)\end{array}\right] \geq 1 - \mathsf{negl}(\lambda),$$

and for all $j \in [n]$ such that $j < i$ for all $i \in S$ it holds that

$$\Pr\left[\begin{array}{c}\mathsf{Comb}(\mathsf{pk}, c, S, \{d_i\}_S) \\ = k\end{array} : \begin{array}{c}(\mathsf{pp}, \mathsf{pk}, \{\mathsf{sk}_i\}_{[n]}, \mathsf{tk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, n, t) \\ (k, c) \xleftarrow{\$} \mathsf{TrEnc}(\mathsf{tk}, j) \\ \forall i \in S\colon d_i \leftarrow \mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)\end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

C Deferred Proofs from Section 6

C.1 Proof of Simulability of Partial Decryptions

Proof of Lemma 6.2. Consider an adversary $\mathcal{A}$ running in $\mathsf{GameSimPD}$ (Fig. 4) instantiated with an arbitrary $\varepsilon \in 1/\mathsf{poly}(\lambda)$ and $m = \lambda$. Let $q \in \mathsf{poly}(\lambda)$ be an upper bound on the number of queries to the random oracle $\mathcal{H}$ made by $\mathcal{A}$. W.l.o.g., assume that, whenever the adversary queries the oracle Chal on input $r_{\mathsf{Enc}} = (\mathsf{pre}, v)$, then $\mathcal{A}$ also issues a query to the random oracle $\mathcal{H}(\mathsf{pre})$.

Consider a Chal query for encryption randomness $r_{\mathsf{Enc}} \in \mathcal{R}$ with $r_{\mathsf{Enc}} = (\mathsf{pre} \in \{0,1\}^\lambda, v \in \mathbb{Z}_p)$. In this case the ciphertext is generated as follows: pre is used to sample m keys from the preprocessing as $(K_1, \ldots, K_m) \leftarrow \mathsf{Samp}(\mathcal{U}; \mathcal{H}(\mathsf{pre}))$, and then the KEM key $k = [vs]_T$ is encrypted under all keys $K_1, \ldots, K_m$, yielding the ciphertext $c = ([v]_2, \{Z_q, \mathsf{idx}_q\}_{q \in [m]})$, where idx_q is the index of K_q in the preprocessing $\mathcal{U}$.

If $b = 0$, the Chal oracle will now generate *normal* partial decryptions on behalf of all parties on the ciphertext c. If, however, $b = 1$, the oracle will instead generate patial decryptions using $\mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$ for all $i \in [n]$. In the latter case, the partial decryptions are always equal to $[vs_i]_T$.

Next, recall that $\mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)$ for a valid ciphertext (i.e, a ciphertext corresponding to randomness $r_{\mathsf{Enc}} \in \mathcal{R}$) also always outputs $d_i = [vs_i]_T$ except if party i is revoked *in all keys* $K_1, \ldots, K_m$, in which case $\mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c)$ outputs $\perp$.

Observe that, as long as at least one of the sampled keys belongs to $\mathcal{U}_{\mathrm{Norm}}$, i.e., if there exists one $q \in [m]$ such that $K_q \in \mathcal{U}_{\mathrm{Norm}}$, then $\mathsf{PDec}(\mathsf{pk}, \mathsf{sk}_i, c) = \mathsf{SimPD}(\mathsf{pk}, i, c, r_{\mathsf{Enc}})$ for all $i \in [n]$. Hence, we can bound the adversary's advantage in guessing b by its ability to find some pre such that $K_q \notin \mathcal{U}_{\mathrm{Norm}}$ for all $q \in [m]$, where $(K_1, \ldots, K_m) \leftarrow \mathsf{Samp}(\mathcal{U}; \mathcal{H}(\mathsf{pre}))$. Since $|\mathcal{U}_{\mathrm{Norm}}| = \frac{1}{2} \cdot |\mathcal{U}|$, the probability that this happens for any given preimage pre is $\leq 2^{-m}$. A union bound over the maximum number of queries q gives us the following bound on $\mathcal{A}$'s advantage

$$\mathsf{Adv}^{\mathsf{SimPD}}_{\mathcal{A},\text{Constr. 2}}(1^\lambda) \leq q \cdot 2^{-m} = \frac{q}{2^\lambda},$$

which is negligible in λ. ∎

C.2 Proof of Indistinguishability

Proof of Lemma 6.3. In IndEmp, the challenger takes control of the tracer, while the adversary controls all n parties.

In the first step of our proof, we must argue that the commitment com to preprocessing randomness r_{Prep} that the tracer publishes does not leak r_{Prep}. This is because r_{Prep} contains the random permutation to shuffle preprocessing keys, and thus would allow the adversary and its decoder to trivially distinguish $\mathcal{U}_{\mathrm{Norm}}$ from $\mathcal{U}_{\mathrm{TR}}[0]$. Let Game_0 be identical to IndEmp when instantiated with Construction 2.

$\underline{\mathsf{Game}_1}$: In Game_1, we make a modification to TrCom. Instead of committing to the preprocessing randomness r_{Prep}, the challenger commits to a constant of equal length 0^ℓ, where ℓ denotes the number of random bits sampled by Prep. More formally, instead of computing $\mathsf{com} \xleftarrow{\$} \Pi_{\mathsf{Com}}.\mathsf{Com}(\mathsf{pp}_{\mathsf{Com}}, r_{\mathsf{Prep}})$, the challenger computes $\mathsf{com} \xleftarrow{\$} \Pi_{\mathsf{Com}}.\mathsf{Com}(\mathsf{pp}_{\mathsf{Com}}, 0^\ell)$.

Claim C.1 For all PPT adversaries $\mathcal{A}$ there exists a PPT reduction $\mathcal{B}$ such that

$$\left|\Pr\left[\mathsf{Game}_{1,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\right] - \Pr\left[\mathsf{Game}_{0,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\right]\right| \leq 2 \cdot \mathsf{Adv}^{\mathrm{hiding}}_{\mathcal{B},\Pi_{\mathsf{Com}}}(1^\lambda).$$

A proof for an almost identical claim can be found in the proof of index hiding (Appendix C.3, Claim C.2).

Since the keys in $\mathcal{U}_{\mathrm{Norm}}$ and $\mathcal{U}_{\mathrm{TR}}[0]$ are generated from the same distribution (i.e., revoke no parties) we can reduce the modified indistinguishability game Game_1 to the statistical problem $\mathsf{KEYS\text{-}IND}$ (Appendix D).

Let $\mathcal{A}$ be a PPT adversary against Game_1, we next show a PPT adversary $\mathcal{B}$ that plays against $\delta\text{−}\mathsf{KEYS\text{-}IND}$ with parameters $L = n + 2$, $d = |\mathcal{U}_{\mathrm{TR}}[0]|$, and $\delta = \frac{\varepsilon}{8}$. We describe a reduction in which the adversary $\mathcal{B}$ uses $\mathcal{A}$ in order to win the game $\delta\text{−}\mathsf{KEYS\text{-}IND}$. In defining the distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ in $\mathsf{KEYS\text{-}IND}$, we choose $p_\alpha = 1/2$ and $p_\beta = 1/(2n+2)$.

Reduction.

- $\mathcal{B}$ receives the global set $[Ld]$.
- $\mathcal{B}$ generates the CRS honestly and simulates TrCom by computing $\mathsf{com} \xleftarrow{\$} \Pi_{\mathsf{Com}}.\mathsf{Com}(\mathsf{pp}_{\mathsf{Com}}, 0^\ell)$. $\mathcal{B}$ forwards the CRS and com to $\mathcal{A}$.

- $\mathcal{A}$ provides $\mathcal{B}$ the randomness used for key generation on behalf of all parties, and $\mathcal{B}$ executes KGen accordingly.
- $\mathcal{B}$ executes the preprocessing phase of Game_1:
  - It generates all keys in $\mathcal{U}_{\text{TR}}[j]$ for every $j \neq 0$.
  - It then samples Ld preprocessing keys that do not revoke any party. We denote this set by U. Note that this is exactly how keys from both $\mathcal{U}_{\text{Norm}}$ and $\mathcal{U}_{\text{TR}}[0]$ are sampled in IndEmp.
  - $\mathcal{B}$ shuffles all preprocessing keys $U \cup \bigcup_{1 \leq j \leq n} \mathcal{U}_{\text{TR}}[j]$, and forwards the resulting preprocessing $\mathcal{U}$ to $\mathcal{A}$.
- $\mathcal{A}$ receives the preprocessing keys and outputs a decoder $D = (D_1, D_2)$.
- $\mathcal{B}$ constructs a decoder D' as follows:
  - On input $q = (v_1, \ldots, v_\lambda)$, where $v_i \in [Ld] \cup \{\perp\}$:
    - For each $i \in [\lambda]$, let

$$K_i \leftarrow \begin{cases} \text{Sample a key uniformly without replacement from } \bigcup_{j>0} \mathcal{U}_{\text{TR}}[j] & \text{if } v_i = \perp, \\ U[v_i] & \text{otherwise.} \end{cases}$$

    - Sample $v \xleftarrow{\$} \mathbb{Z}_p$, $(k_0, c) \xleftarrow{\$} \mathsf{enc}(\mathsf{CRS}, \{K_i, \mathsf{idx}_i\}_{i \in [\lambda]}; v)$, and $k_1 \xleftarrow{\$} \mathcal{K}$. Here, idx_i denotes the index of K_i in $\mathcal{U}$.
    - Sample $b \xleftarrow{\$} \{0,1\}$ and compute $(\mathsf{st}, \mathcal{I}_{\mathsf{pd}}) \leftarrow D_1(k_b, c)$.
    - Compute simulated decryption shares

$$\{d_i\}_{i \in \mathcal{I}_{\mathsf{pd}}} \leftarrow \mathsf{SimPDs}(\mathcal{I}_{\mathsf{pd}}, \rho, \mathsf{pk}, c, v).$$

    - Let $b' \leftarrow D_2(\mathsf{st}, \{d_i\}_{i \in \mathcal{I}_{\mathsf{pd}}})$.
    - Output 1 if and only if $b = b'$.

Analysis. We next argue that the above reduction perfectly simulates Game_1 for $\mathcal{A}$.

- All preprocessing keys are generated exactly as in IndEmp, so these keys are distributed identically to the IndEmp.
- The encryption is done through the same enc method used in Enc and TrEnc.
- The keys used to generate the ciphertexts in TrEnc are distributed identically to those in TrEnc in IndEmp. To see this, observe that the distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ over the sets U_0 and U_1 (chosen by the challenger in $\mathsf{KEYS\text{-}IND}$) correspond exactly to the distributions used in Enc and $\mathsf{TrEnc}(0)$, respectively. In particular, we interpret U_0 and U_1 as inducing a random partition of the sets $\mathcal{U}_{\text{Norm}}$ and $\mathcal{U}_{\text{TR}}[0]$ in IndEmp. That is, letting $U = \mathcal{U}_{\text{Norm}} \cup \mathcal{U}_{\text{TR}}[0]$, we have $\mathcal{U}_{\text{Norm}} = \{U[j]\}_{j \in U_0}$ and $\mathcal{U}_{\text{TR}}[0] = \{U[j]\}_{j \in U_1}$. Therefore, given a sample of λ values $v_1, \ldots, v_\lambda$, replacing each value $v_i \neq \perp$ with the corresponding key $U[v_i]$ yields the desired distribution over U. Furthermore, each occurrence of $\perp$ is replaced with a key sampled uniformly from $U' = \bigcup_{j \in [n]} \mathcal{U}_{\text{TR}}[j]$. Since $\perp$ appears with the same probability in both $\mathcal{D}_0$ and $\mathcal{D}_1$, corresponding to sampling a key from U', the reduction perfectly simulates the distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ over $U \cup U'$ as in IndEmp. Consequently, it perfectly simulates Enc and $\mathsf{TrEnc}(0)$.
- A subtle point is that we provide simulated partial decryptions to the decoder D. Observe, however, that the reduction $\mathcal{B}$ can simulate these values perfectly from its view. Indeed, in SimPD, the partial decryption of party i is given by $[s_i v]_T$, where s_i is the share of s stored in the CRS, and v is the randomness used during encryption, which is sampled by $\mathcal{B}$. In particular, this simulation does not require knowledge of the partition of U, which remains unknown to $\mathcal{B}$. Intuitively, the simulated partial decryptions do not depend on the keys at all, and therefore provide no advantage to the adversary in distinguishing between Enc and $\mathsf{TrEnc}(0)$.

Therefore, if the decoder D can distinguish between Enc and $\mathsf{TrEnc}(0)$, then the constructed decoder D' distinguishes between $\mathcal{D}_0$ and $\mathcal{D}_1$ with the same probability. Indeed, when D' is evaluated in the $\mathsf{KEYS\text{-}IND}$ game, the case $b = 0$ essentially simulates the number of times D correctly distinguishes Enc ciphertexts, while $b = 1$ simulates the number of times D correctly distinguishes $\mathsf{TrEnc}(0)$ ciphertexts. More concretely, let a and b denote the empirical success rates computed in IndEmp as in Fig. 5, for Enc and $\mathsf{TrEnc}(0)$, respectively. Let p_0 and p_1 denote the corresponding counters in the δ-$\mathsf{KEYS\text{-}IND}$ experiment for the decoder D'.

By the perfect simulation above, samples from $\mathcal{D}_0$ correspond to Enc ciphertexts, while samples from $\mathcal{D}_1$ correspond to $\mathsf{TrEnc}(0)$ ciphertexts. Consequently,

$$|b - a| = \left| \frac{p_1}{Q} - \frac{p_0}{Q} \right|.$$

Therefore, if the empirical test in IndEmp yields $|b - a| \geq \delta$, then the empirical test in δ-$\mathsf{KEYS\text{-}IND}$ yields $\left| \frac{p_1}{Q} - \frac{p_0}{Q} \right| \geq \delta$.

To conclude the proof, from Lemma D.1 and $L = n + 2$, it holds that

$$\left| \frac{p_1}{Q} - \frac{p_0}{Q} \right| = |b - a| \leq \frac{\sqrt{8n + 16}\lambda^{3/2}}{\sqrt{d}} + 2\sqrt{\frac{\lambda}{2Q}}.$$

Thus, choosing $Q \geq \frac{8\lambda}{\delta^2}$ and $d \geq \frac{32n\lambda^3}{\delta^2}$, ensures that $|b - a| \leq \delta$ except for a negligible probability in λ. In particular, setting $\delta = \varepsilon/8$, we obtain $d = 2048\lambda^3 n/\varepsilon^2$ and $Q = 512\lambda/\varepsilon^2$. Hence, we get that the construction achieves $(\varepsilon/8, t - 1, \frac{512\lambda}{\epsilon^2})$-indistinguishability. ∎

C.3 Proof of Index Hiding

Proof of Lemma 6.4. When instantiating the IHEmp game (Fig. 5) with Construction 2, the adversary $\mathcal{A}$ first specifies the parameters n and t, as well as a challenge index i^*. Then, we execute the Setup protocol, where the adversary controls all parties $\{\mathbb{P}_i\}_{i \in [n] \setminus \{i^*\}}$ except party $\mathbb{P}_{i^*}$, and the challenger controls the CRS, tracer, and party $\mathbb{P}_{i^*}$. In particular, this means that the challenger first computes the CRS by executing $\mathsf{CRSGen}(1^\lambda, n, t)$, and then the commitment com as well as randomness r_{Prep} by executing $\mathsf{TrCom}(1^\lambda, n, t, \mathsf{CRS})$ on behalf of the tracer. The challenger then executes key generation on behalf of party i^*, computing $(\mathsf{pk}_{i^*}, \mathsf{sk}_{i^*}) \xleftarrow{\$} \mathsf{KGen}(1^\lambda, \mathsf{CRS}, i^*)$. The challenger sends $(\mathsf{CRS}, \mathsf{com}, \mathsf{pk}_{i^*})$ to the adversary, who picks randomness $\{\alpha_i^{(q)}\}_{q \in [m]}$ for key generation on behalf of all remaining parties $i \in [n] \setminus \{i^*\}$ and forwards it back to the challenger. The challenger then computes $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KGen}(1^\lambda, \mathsf{CRS}, i; \{\alpha_i^{(q)}\}_{q \in [m]})$ for all $i \in [n] \setminus \{i^*\}$ and computes the preprocessing $(\mathcal{U}, \mathsf{tk}) \leftarrow \mathsf{Prep}(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i \in [n]}; r_{\mathsf{Prep}})$, forwarding $\mathcal{U}$ to the adversary, who outputs a decoder $D = (D_1, D_2)$. Finally, the challenger performs the empirical test on the decoder, computing a and b as the fraction of times the decoder decrypts correctly for $\mathsf{TrEnc}(\mathsf{tk}, i^* - 1)$ and $\mathsf{TrEnc}(\mathsf{tk}, i^*)$ across Q repetitions, respectively. Finally, the IHEmp game outputs 1, iff $|a - b| \geq \delta$.

We next prove Lemma 6.4 in two steps. First, we show that, under the XDH assumption, and if the commitment is hiding, then $\mathcal{U}_{\mathrm{TR}}[i^* - 1]$ is computationally indistinguishable from $\mathcal{U}_{\mathrm{TR}}[i^*]$ by adversaries who do not know sk_{i^*}. Then, we show by a reduction to the statistical problem $\mathsf{KEYS\text{-}IND}$ that it is infeasible to construct a decoder that distinguishes $\mathsf{TrEnc}(i^* - 1)$ from $\mathsf{TrEnc}(i^*)$ with sufficient advantage merely by the distribution over the indices of preprocessing keys chosen in the respective ciphertexts.

Proving that $\mathcal{U}_{\mathbf{tr}}[i^*] \approx_c \mathcal{U}_{\mathbf{tr}}[i^* - 1]$. We introduce a series of game hops, such that in the final game, the keys in $\mathcal{U}_{\mathrm{TR}}[i^*]$ are sampled identically to those in $\mathcal{U}_{\mathrm{TR}}[i^* - 1]$. Let Game_0 be identical to IHEmp when instantiated with Construction 2.

First, we argue that the commitment com to the preprocessing randomness r_{Rand} does not leak information about the keys.

$\underline{\mathsf{Game}_1}$: In Game_1, we make a modification to TrCom. Instead of committing to the preprocessing randomness r_{Prep}, the challenger commits to a constant of equal length 0^ℓ, where ℓ denotes the number of random bits sampled by Prep. More formally, instead of computing $\mathsf{com} \xleftarrow{\$} \Pi_{\mathsf{Com}}.\mathsf{Com}(\mathsf{pp}_{\mathsf{Com}}, r_{\mathsf{Prep}})$, the challenger computes $\mathsf{com} \xleftarrow{\$} \Pi_{\mathsf{Com}}.\mathsf{Com}(\mathsf{pp}_{\mathsf{Com}}, 0^\ell)$.

Claim C.2 For all PPT adversaries $\mathcal{A}$ there exists a PPT reduction $\mathcal{B}$ such that

$$\left| \Pr\left[\mathsf{Game}_{1,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\right] - \Pr\left[\mathsf{Game}_{0,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\right] \right| \leq 2 \cdot \mathsf{Adv}^{\mathrm{hiding}}_{\mathcal{B}, \Pi_{\mathsf{Com}}}(1^\lambda).$$

Proof of Claim C.2. We prove the claim by a straightforward reduction to hiding of Π_{Com}. After receiving $\mathsf{pp}_{\mathsf{Com}}$ from its challenger, the reduction simulates both Game_1 and Game_0 to $\mathcal{A}$. When $\mathcal{B}$ needs to run TrCom, it samples r_{Prep}, and forwards $(r_{\mathsf{Prep}}, 0^\ell)$ to its challenger. Note that $|r_{\mathsf{Prep}}| = |0^\ell| = \ell$. As a result, the challenger in the hiding experiment returns com, which $\mathcal{B}$ forwards to $\mathcal{A}$. $\mathcal{B}$ continues to simulate as required by both Game_0 and Game_1, and in the end, outputs as either game would, i.e., 1, iff $|a-b| \geq \delta$ and 0 otherwise.

Analysis. Let b denote the internal bit of the hiding challenger, and b' be $\mathcal{B}$'s output. Observe that $\mathcal{B}$ perfectly simulates Game_b to $\mathcal{A}$, hence we get that

$$\begin{aligned}
\mathsf{Adv}^{\text{comp. hiding}}_{\mathcal{B},\Pi_{\mathsf{Com}}} &= \left|\Pr[b'=b] - \frac{1}{2}\right| \\
&= \left|\frac{1}{2}\cdot\Pr[b'=0 \mid b=0] + \frac{1}{2}\cdot\Pr[b'=1 \mid b=1] - \frac{1}{2}\right| \\
&= \left|\frac{1}{2}\left(1 - \Pr[b'=1 \mid b=0] + \Pr[b'=1 \mid b=1]\right) - \frac{1}{2}\right| \\
&= \left|\frac{1}{2}\left(\Pr[b'=1 \mid b=1] - \Pr[b'=1 \mid b=0]\right)\right| \\
&= \frac{1}{2}\cdot\left|\Pr\left[\mathsf{Game}_{1,\mathcal{A}}(1^\lambda,\delta,\rho,Q)=1\right] - \Pr\left[\mathsf{Game}_{0,\mathcal{A}}(1^\lambda,\delta,\rho,Q)=1\right]\right|.
\end{aligned}$$

■

Next, we modify how $\mathcal{U}_{\text{TR}}[i^*-1]$ is sampled.

$\underline{\mathsf{Game}_2}$: In Game_2, we modify how the Prep procedure samples the keys of $\mathcal{U}_{\text{TR}}[i^*-1]$. In particular, for all $\ell \in [d]$, instead of generating

$$\mathcal{U}_{\text{TR}}[i^*-1,\ell] \xleftarrow{\$} \mathsf{Rand}(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i\in[n]}, \mathcal{R}_{i^*-1}),$$

we generate the preprocessing keys as

$$\mathcal{U}_{\text{TR}}[i^*-1,\ell] \xleftarrow{\$} \mathsf{Rand}(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i\in[n]}, \mathcal{R}_{i^*}).$$

That is, we sample them identically to $\mathcal{U}_{\text{TR}}[i^*]$, revoking the set $\mathcal{R}_{i^*}$.

Claim C.3 Let $n_{\max} \in \mathsf{poly}(\lambda)$. For all PPT adversaries $\mathcal{A}$ that initially output $n \leq n_{\max}$ there exists a PPT reduction $\mathcal{B}$ such that

$$\left|\Pr\left[\mathsf{Game}_{2,\mathcal{A}}(1^\lambda,\delta,\rho,Q)=1\right] - \Pr[\mathsf{Game}_{1,\mathcal{A}}(1^\lambda,\delta,\rho,Q)=1]\right| \leq d_{\max}\cdot\mathsf{Adv}^{\mathsf{XDH}}_{\mathcal{B},\mathcal{G}}(1^\lambda),$$

where $d_{\max} = 1024\cdot\lambda m^2 n_{\max}^2/\varepsilon^2$.

Proof of Claim C.3. W.l.o.g., assume that the adversary $\mathcal{A}$ initially outputs $n = n_{\max}$ and thus $|\mathcal{U}_{\text{TR}}[i]| = d_{\max} = d$. We prove the claim through a hybrid argument. Let $\mathcal{H}_0$ be identical to Game_1, and for $0 < u \leq d$, let $\mathcal{H}_u$ be the game where we replace the first u keys of $\mathcal{U}_{\text{TR}}[i^*-1]$ with keys that revoke $\mathcal{R}_{i^*}$. Observe that $\mathcal{H}_d$ is identical to Game_2.

We next bound $\mathcal{A}$'s success probability in subsequent hybrids $\mathcal{H}_{u-1}$ and $\mathcal{H}_u$ by a PPT reduction $\mathcal{B}$ to the XDH assumption (Asmpt. 1). First, $\mathcal{B}$ receives an XDH challenge $[x]_1, [y]_1, T$, where either $T = [xy]_1$ or $T = [z]_1$ for a random z. The reduction $\mathcal{B}$ samples a random target $u \xleftarrow{\$} [d]$ and then runs $\mathcal{A}(1^\lambda)$ to receive $i^* \in [n]$. Then, the reduction samples the CRS and the commitment $\mathsf{com} \leftarrow \Pi_{\mathsf{Com}}.\mathsf{Com}(\mathsf{pp}_{\mathsf{Com}}, 0^\ell)$ and forwards them to $\mathcal{A}$.

Simulating pk_{i^*} Next, the reduction embeds the XDH challenge into the public key of party i^* by sampling $w^{(q)} \xleftarrow{\$} \mathbb{Z}_p^*$ and simulating key generation such that $\alpha_{i^*}^{(q)} \equiv x\cdot w^{(q)}$. To do so, set

$$\mathsf{pk}_{i^*} \leftarrow \left(\{w^{(q)} c_{i^*}\cdot[x]_1\}_{q\in[m]}, \{w^{(q)} c_{i^*}/c_j\cdot[x]_1\}_{q\in[m],j\in[n]\setminus\{i^*\}}\right),$$

which $\mathcal{B}$ can do as it knows the cross terms from the CRS as field elements. $\mathcal{B}$ forwards pk_{i^*} to $\mathcal{A}$ and receives $\{\alpha_i^{(q)}\}_{q\in[m]}$ for all other parties $i \neq i^*$ as a response, allowing it to compute $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KGen}(1^\lambda, \mathsf{CRS}, i; \{\alpha_i^{(q)}\}_{q\in[m]})$ for all $i \neq i^*$.

Simulating the preprocessing $\mathcal{U}$. Next, $\mathcal{B}$ simulates the preprocessing as follows. $\mathcal{B}$ generates all preprocessing keys in $\mathcal{U}_{\text{Norm}}$ and $\mathcal{U}_{\text{TR}}[j]$ for $j \neq i^*-1$ honestly by rerandomizing according to protocol. To generate the preprocessing keys for $\mathcal{U}_{\text{TR}}[i^*-1,q]$ for $\ell \in [d]$, $\mathcal{B}$ does the following:

- **For** $\ell < u$, generate keys that revoke the set $\mathcal{R}_{i^*}$ by running the rerandomization algorithm $\mathsf{Rand}(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i\in[n]}, \mathcal{R}_{i^*})$.
- **For** $\ell > u$, generate keys that revoke the set $\mathcal{R}_{i^*-1}$ by running the rerandomization algorithm $\mathsf{Rand}(\mathsf{CRS}, \{\mathsf{pk}_i\}_{i\in[n]}, \mathcal{R}_{i^*-1})$.
- **For** $\ell = u$, embed the XDH challenge into the preprocessing key such that $\beta \equiv y$. Set $[\beta]_1 \leftarrow [y]_1$. For all revoked parties $i \in [i^*-1]$, rerandomization uses unique randomness. Thus, $\mathcal{B}$ samples $r_i \xleftarrow{\$} \mathbb{Z}_p^*$ and computes

$$A_i^{(q)} \leftarrow \left[c_i \alpha_i^{(q)} r_i\right]_1 \text{ for all } q \in [m].$$

For all unrevoked parties $i \in [i^*+1, n]$, rerandomization happens with respect to β, and thus $\mathcal{B}$ computes

$$A_i^{(q)} \leftarrow c_i \alpha_i^{(q)} \cdot [y]_1 \text{ for all } q \in [m].$$

For i^*, compute

$$A_{i^*}^{(q)} \leftarrow w^{(q)} c_{i^*} \cdot T \text{ for all } q \in [m].$$

Next, compute $B_i^{(q)}$ for all $i \in [n] \setminus \{i^*\}$ and all $q \in [m]$ as

$$B_i^{(q)} \leftarrow \frac{w^{(q)} c_{i^*}}{c_i} \cdot T + \sum_{j\in[i^*-1]\setminus\{i\}} \left[\alpha_j^{(q)} r_j c_j / c_i\right]_1 + \sum_{i\in[i^*+1,n]\setminus\{i\}} \frac{\alpha_j^{(q)} c_j}{c_i} \cdot [y]_1 ,$$

as well as

$$B_{i^*}^{(q)} \leftarrow \sum_{j\in[i^*-1]} \left[\alpha_j^{(q)} r_j c_j / c_{i^*}\right]_1 + \sum_{i\in[i^*+1,n]} \alpha_j^{(q)} c_j / c_{i^*} \cdot [y]_1 .$$

Essentially, this serves to achieve the following: If $T = [xy]_1$, then party i^* is rerandomized by β (i.e., unrevoked) in $\mathcal{U}_{\text{TR}}[i^*-1, u]$, and thus $\mathcal{U}_{\text{TR}}[i^*-1]$ is distributed as in $\mathcal{H}_{u-1}$. If, however, $T = [z]_1$ for a uniform random z, then the above is equivalent to rerandomizing party i^* with $r_i \equiv z/x$, which means that party i^* is revoked in $\mathcal{U}_{\text{TR}}[i^*-1, u]$, and thus $\mathcal{U}_{\text{TR}}[i^*-1]$ is distributed as in $\mathcal{H}_u$.

The reduction $\mathcal{B}$ then forwards the preprocessing to $\mathcal{A}$ and receives a decoder $D = (D_1, D_2)$ in return. It computes a and b as mandated by either hybrid by testing the decoder. Finally, $\mathcal{B}$ outputs 1, iff $|a-b| \geq \delta$, i.e., if $\mathcal{A}$ would win either hybrid, and 0 otherwise.

Analysis. Suppose that $\mathcal{B}$ initialy samples $u \in [d]$. If in the XDH challenge it holds that $T = [xy]_1$, $\mathcal{B}$ perfectly simulates $\mathcal{H}_{u-1,\mathcal{A}}(1^\lambda, \delta, \rho, Q)$. If, however, $T = [z]_1$, then $\mathcal{B}$ perfectly simulates $\mathcal{H}_{u,\mathcal{A}}(1^\lambda, \delta, \rho, Q)$. Hence, we get that

$$\begin{aligned}
&\left|\Pr\left[\mathcal{H}_{u,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\right] - \Pr\left[\mathcal{H}_{u-1,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\right]\right| \\
&= \left|\Pr\left[\mathcal{B}(1^\lambda, E, [x]_1, [y]_1, [z]_1) = 1 \mid \mathcal{B} \text{ picks } u\right] - \Pr\left[\mathcal{B}(1^\lambda, E, [x]_1, [y]_1, [xy]_1) = 1 \mid \mathcal{B} \text{ picks } u\right]\right| .
\end{aligned}$$

Finally, we conclude that

$$\begin{aligned}
&\left|\Pr\left[\mathsf{Game}_{2,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\right] - \Pr[\mathsf{Game}_{1,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1]\right| \\
&= \left|\Pr\left[\mathcal{H}_{d,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\right] - \Pr[\mathcal{H}_{0,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1]\right| \\
&= \left|\sum_{j=1}^{d} \Pr\left[\mathcal{H}_{j,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\right] - \Pr\left[\mathcal{H}_{j-1,\mathcal{A}}(1^\lambda, \delta, \rho, Q) = 1\right]\right| \\
&= \left|\sum_{j=1}^{d} \Pr\left[\mathcal{B}(1^\lambda, E, [x]_1, [y]_1, [z]_1) = 1 \mid \mathcal{B}.u = j\right] - \Pr\left[\mathcal{B}(1^\lambda, E, [x]_1, [y]_1, [xy]_1) = 1 \mid \mathcal{B}.u = j\right]\right| \\
&= \left|\sum_{j=1}^{d} \frac{\Pr\left[\mathcal{B}(1^\lambda, E, [x]_1, [y]_1, [z]_1) = 1 \wedge \mathcal{B}.u = j\right] - \Pr\left[\mathcal{B}(1^\lambda, E, [x]_1, [y]_1, [xy]_1) = 1 \wedge \mathcal{B}.u = j\right]}{\Pr[\mathcal{B}.u = j]}\right| \\
&= d \cdot \left|\Pr\left[\mathcal{B}(1^\lambda, E, [x]_1, [y]_1, [z]_1) = 1\right] - \Pr\left[\mathcal{B}(1^\lambda, E, [x]_1, [y]_1, [xy]_1) = 1\right]\right| \\
&= d \cdot \mathsf{Adv}^{\mathsf{XDH}}_{\mathcal{B},\mathcal{G}}(1^\lambda),
\end{aligned}$$

which proves the claim. ∎

Reduction to a Statistical Problem.

Since the keys $\mathcal{U}_{\text{TR}}[i^*-1]$ and $\mathcal{U}_{\text{TR}}[i^*]$ are computationally indistinguishable to any adversary that does not know the secret key sk_i, the index-hiding game can be reduced to a statistical problem KEYS-IND (Section D).

The reduction. We next show that an adversary against KEYS-IND can use an adversary against Game_2. That is, we prove the following claim.

Claim C.4 For every PPT adversary $\mathcal{A}$ against Game_2 there exists a PPT adversary $\mathcal{B}$ against $\delta-$KEYS-IND such that

$$\mathsf{Adv}^{\mathsf{Game}_2}_{\mathcal{A}} \leq \mathsf{Adv}^{\delta-\mathsf{KEYS\text{-}IND}}_{\mathcal{B}}.$$

Proof. Let $\mathcal{A}$ be a PPT adversary against Game_2, we next show a PPT adversary $\mathcal{B}$ that plays against $\delta-$KEYS-IND with parameters $L=2$, $d=|\mathcal{U}_{\text{TR}}[i^*]|$, and $\delta=\frac{\epsilon}{4n}$. We describe a reduction in which the adversary $\mathcal{B}$ uses $\mathcal{A}$ in order to win the game $\delta-$KEYS-IND. In defining the distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ in KEYS-IND, we choose $p_\alpha=\frac{1}{2}+\frac{i^*}{2n+2}$ and $p_\beta=\frac{1}{2n+2}$ (this is the same distribution of sampling keys in $\mathsf{TrEnc}(i^*)$ and $\mathsf{TrEnc}(i^*-1)$).

Reduction.

- $\mathcal{B}$ receives the global set $[2d]$.
- $\mathcal{B}$ generates the CRS honestly and simulates TrCom by computing $\mathsf{com} \xleftarrow{\$} \Pi_{\mathsf{Com}}.\mathsf{Com}(\mathsf{pp}_{\mathsf{Com}}, 0^\ell)$. $\mathcal{B}$ forwards the CRS and com to $\mathcal{A}$.
- $\mathcal{A}$ provides $\mathcal{B}$ with the challenge index i^* together with all public keys. Furthermore, it provides all secret keys sk_j for $j \neq i^*$.
- $\mathcal{B}$ executes the preprocessing phase of Game_2:
  - It generates all keys in $\mathcal{U}_{\text{Norm}}$ and $\mathcal{U}_{\text{TR}}[j]$ for every $j \neq i^*-1, i^*$.
  - It then samples the keys in $\mathcal{U}_{\text{TR}}[i^*-1]$ and $\mathcal{U}_{\text{TR}}[i^*]$ so as to revoke the first i^* parties. In particular, it generates a set U of $2d$ keys corresponding to this revocation. Note that this is exactly how keys from both $\mathcal{U}_{\text{TR}}[i^*-1]$ and $\mathcal{U}_{\text{TR}}[i^*]$ are sampled in Game_2.
  - $\mathcal{B}$ shuffles all preprocessing keys $U \cup \mathcal{U}_{\text{Norm}} \cup \bigcup \mathcal{U}_{\text{TR}}[j]_{j\neq i^*-1,i^*}$, and forwards the resulting preprocessing $\mathcal{U}$ to $\mathcal{A}$.
- $\mathcal{A}$ receives the preprocessing keys and outputs a decoder $D=(D_1,D_2)$.
- $\mathcal{B}$ constructs a decoder D' as follows:
  - On input $q=(v_1,\ldots,v_\lambda)$, where $v_i \in [2d]\cup\{\perp\}$:
    - For each $i \in [\lambda]$, define

$$K_i := \begin{cases} \text{Sample a key uniformly without replacement from } \bigcup_{j>i^*} \mathcal{U}_{\text{TR}}[j] & \text{if } v_i=\perp, \\ U[v_i] & \text{otherwise.} \end{cases}$$

    - Sample $v \xleftarrow{\$} \mathbb{Z}_p$, $(k_0,c) \xleftarrow{\$} \mathsf{enc}(\mathsf{CRS}, \{K_i, \mathsf{idx}_i\}_{i\in[\lambda]}; v)$ and $k_1 \xleftarrow{\$} \mathcal{K}$. Here, idx_i denotes the index of K_i in $\mathcal{U}$.
    - Sample $b \xleftarrow{\$} \{0,1\}$ and compute $(\mathsf{st}, \mathcal{I}_{\mathsf{pd}}) \leftarrow D_1(k_b, c)$.
    - Compute simulated decryption shares

$$\{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}} \leftarrow \mathsf{SimPDs}(\mathcal{I}_{\mathsf{pd}}, \rho, \mathsf{pk}, c, v).$$

    - Let $b' \leftarrow D_2(\mathsf{st}, \{d_i\}_{i\in\mathcal{I}_{\mathsf{pd}}})$.
    - Output 1 if and only if $b=b'$.

Analysis. We argue that the above reduction perfectly simulates Game_2 for $\mathcal{A}$. This is follows due to the following:

- All preprocessing keys are generated exactly as in Game_2, so these keys are distributed identically to the Game_2.

- The encryption is done through the same enc method used in Enc and TrEnc.
- The keys used to generate the ciphertexts in TrEnc are distributed identically to those in TrEnc in Game_2. To see this, observe that the distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ over the sets U_0 and U_1 (chosen by the challenger in $\mathsf{KEYS\text{-}IND}$) correspond exactly to the distributions used in $\mathsf{TrEnc}(i^*-1)$ and $\mathsf{TrEnc}(i^*)$, respectively. In particular, we interpret U_0 and U_1 as inducing a random partition of the sets $\mathcal{U}_{\mathrm{TR}}[i^*-1]$ and $\mathcal{U}_{\mathrm{TR}}[i^*]$ in $\mathsf{KEYS\text{-}IND}$. That is, letting $U = \mathcal{U}_{\mathrm{TR}}[i^*-1] \cup \mathcal{U}_{\mathrm{TR}}[i^*]$, we have $\mathcal{U}_{\mathrm{TR}}[i^*-1] = \{U[j]\}_{j\in U_0}$ and $\mathcal{U}_{\mathrm{TR}}[i^*] = \{U[j]\}_{j\in U_1}$. Therefore, given a sample of λ values $v_1, \ldots, v_\lambda$, replacing each value $v_i \neq \perp$ with the corresponding key $U[v_i]$ yields the desired distribution over U. Furthermore, each occurrence of $\perp$ is replaced with a key sampled uniformly from $U' = \bigcup (\mathcal{U}_{\mathrm{TR}}[j])_{j\in[i^*+1,\ldots,n]}$. Since $\perp$ appears with the same probability in both $\mathcal{D}_0$ and $\mathcal{D}_1$, corresponding to sampling a key from U', the reduction perfectly simulates the distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ over $U \cup U'$ as in Game_2.
- A subtle point is that we now provide simulated partial decryptions to the decoder D. Observe, however, the reduction $\mathcal{B}$ can simulate these values perfectly from its view. Indeed, in SimPD, the partial decryption of party i is given by $[s_i v]_T$, where s_i is the share of s stored in the CRS, and v is the randomness used during encryption, which is sampled by $\mathcal{B}$. In particular, this simulation does not require knowledge of the partition of U, which remains unknown to $\mathcal{B}$. Intuitively, the simulated partial decryptions do not depend on the keys at all, and therefore provide no advantage to the adversary in distinguishing between $\mathsf{TrEnc}(i^*-1)$ and $\mathsf{TrEnc}(i^*)$.

Therefore, if the decoder D can distinguish between $\mathsf{TrEnc}(i^*-1)$ and $\mathsf{TrEnc}(i^*)$, then the constructed decoder D' distinguishes between $\mathcal{D}_0$ and $\mathcal{D}_1$ with the same probability.

Let a and b denote the empirical success rates computed in IHEmp as in Fig. 5 for $\mathsf{TrEnc}(i^*-1)$ and $\mathsf{TrEnc}(i^*)$, respectively. Furthermore, let p_0 and p_1 denote the corresponding counters in the δ-$\mathsf{KEYS\text{-}IND}$ experiment for D'. By the perfect simulation above, samples from $\mathcal{D}_0$ correspond to $\mathsf{TrEnc}(i^*-1)$ ciphertexts, while samples from $\mathcal{D}_1$ correspond to $\mathsf{TrEnc}(i^*)$ ciphertexts. Thus,

$$|b-a| = \left|\frac{p_1}{Q} - \frac{p_0}{Q}\right|.$$

Thus, if the empirical test in IHEmp yields $|b-a| \geq \delta$, then the empirical test in δ-$\mathsf{KEYS\text{-}IND}$ also yields $\left|\frac{p_1}{Q} - \frac{p_0}{Q}\right| \geq \delta$.

Hence, any advantage of $\mathcal{A}$ in Game_2 directly translates into an advantage of $\mathcal{B}$ in δ-$\mathsf{KEYS\text{-}IND}$, which proves Claim C.4. ∎

To conclude the proof of index hiding, by Claim C.4 and Lemma D.1, and since $L = 2$, it holds that

$$|b-a| = \left|\frac{p_1}{Q} - \frac{p_0}{Q}\right| \leq \frac{\sqrt{16}\,\lambda^{3/2}}{\sqrt{d}} + 2\sqrt{\frac{\lambda}{2Q}}$$

except with negligible probability in λ. Thus, choosing $Q \geq \frac{8\lambda}{\delta^2}$ and $d \geq \frac{64\lambda^3}{\delta^2}$ ensures that $|b-a| \leq \delta$ except with negligible probability in λ. In particular, for $\delta = \frac{\epsilon}{4n}$, we obtain $d = 2^{10}\frac{\lambda^3 n^2}{\epsilon^2}$ and $Q \geq 2^7\frac{\lambda n^2}{\epsilon^2}$. Thus, we achieve $\left(\frac{\epsilon}{4n}, t-1, 3\cdot 2^7\frac{n^2\lambda}{\epsilon^2}\right)$-index hiding.

∎

C.4 Proof of Message Privacy

Next, we can proceed with the message privacy proof.

Proof of Lemma 6.6. We prove $(t-1)$-message privacy by a reduction to the (t,n)-$\mathsf{Int\text{-}EBDDH}$ assumption, but, similar to index hiding, we must first assume that the commitment does not leak the randomness for preprocessing r_{Prep} to the adversary. Let Game_0 be the MP game, instantiated with Construction 2.

$\underline{\mathsf{Game}_1}$: In Game_1, we make a modification to TrCom. Instead of committing to the preprocessing randomness r_{Prep}, the challenger commits to a constant of equal length 0^ℓ, where ℓ denotes the number of random bits sampled by Prep. More formally, instead of computing $\mathsf{com} \xleftarrow{\$} \Pi_{\mathsf{Com}}.\mathsf{Com}(\mathsf{pp}_{\mathsf{Com}}, r_{\mathsf{Prep}})$, the challenger computes $\mathsf{com} \xleftarrow{\$} \Pi_{\mathsf{Com}}.\mathsf{Com}(\mathsf{pp}_{\mathsf{Com}}, 0^\ell)$.

Claim C.5 For all PPT adversaries $\mathcal{A}$ there exists a PPT reduction $\mathcal{B}$ such that

$$\left|\Pr\left[\mathsf{Game}_{1,\mathcal{A}}(1^\lambda, t-1) = 1\right] - \Pr\left[\mathsf{Game}_{0,\mathcal{A}}(1^\lambda, t-1) = 1\right]\right| \leq 2\cdot \mathsf{Adv}^{\mathrm{hiding}}_{\mathcal{B},\Pi_{\mathsf{Com}}}(1^\lambda).$$

The proof of Claim C.5 is almost identical to that of Claim C.2. We will proceed by a reduction from Game_1 to the hardness of the (t,n)-Int-EBDDH assumption.

Let $\mathcal{A}$ be any PPT adversary against Game_1. W.l.o.g., assume that $\mathcal{A}$ initially outputs (n,t) such that $n \geq t > 0$. We construct a reduction $\mathcal{B}$ against the $\mathsf{Int\text{-}EBDDH}(1^\lambda, n, t)$ game (Fig. 10).

$\mathcal{B}$ initially receives the tuple $(\mathsf{view}_1, \mathsf{view}_2, \mathsf{view}_T, \mathsf{chal})$, where

$$\mathsf{view}_1 = \left(\left\{ [c_i]_1, [vc_i]_1, \left[\sum_{\ell \in S} \lambda_{\ell,S}^{(i)} y_\ell c_i\right]_1 \right\}_{i\in[n]}, \left\{ [c_i/c_j]_1, \left[\sum_{\ell \in S} \lambda_{\ell,S}^{(i)} y_\ell c_i/c_j\right]_1 \right\}_{i\neq j\in[n]} \right)$$
$$\mathsf{view}_2 = \left([v]_2, \{[1/c_i]_2\}_{i\in[n]}\right)$$
$$\mathsf{view}_T = \left([y_0]_T\right),$$

where $S = \{0,\ldots,t-1\}$, y_ℓ is uniformly random from $\mathbb{Z}_p$ for all $\ell \in S$, c_i is uniformly random from $\mathbb{Z}_p^*$ for all $i \in [n]$, and v is uniformly random from $\mathbb{Z}_p$. If the internal bit b of Int-EBDDH is 0, then $\mathsf{chal} = [y_0 \cdot v]_T$, and if $b = 1$, then chal is a random element from $\mathbb{G}_T$. $\mathcal{B}$ now embeds the (t,n)-Int-EBDDH tuple into its simulation of Game_1 to $\mathcal{A}$:

Simulating the CRS. $\mathcal{B}$ simulates the CRS such that $s \equiv y_0$, and $s_\ell \equiv y_\ell$ for all $\ell \in [t-1]$. More formally, $\mathcal{B}$ assembles the CRS as follows (elements from the (t,n)-Int-EBDDH tuple are highlighted in gray):

- $[s]_T \leftarrow [y_0]_T$.
- For all $i \in [n]$:
  - $[c_i]_1 \leftarrow [c_i]_1$.
  - $[c_i s_i]_1 \leftarrow \left[\sum_{\ell\in S} \lambda_{\ell,S}^{(i)} y_\ell c_i\right]_1$.
  - $\left[\sum_{j\in[n]\setminus\{i\}} s_i c_i/c_j\right]_1 \leftarrow \sum_{j\in[n]\setminus\{i\}} \left[\sum_{\ell\in S} \lambda_{\ell,S}^{(i)} y_\ell c_i/c_j\right]_1$.
  - $[1/c_i]_2 \leftarrow [1/c_i]_2$.
  - For all $j \in [n]\setminus\{i\}$: $[c_i/c_j]_1 \leftarrow [c_i/c_j]_1$.

$\mathcal{B}$ forwards the CRS to $\mathcal{A}$, alongside a commitment com to $\mathbf{0}$ to simulate CRSGen and TrCom.

Since, in message privacy, all parties are corrupted by the adversary, $\mathcal{B}$ receives the randomness for key generation $\{\alpha_i^{(q)}\}_{q\in[m]}$ on behalf of all parties $i \in [n]$ from the internal adversary $\mathcal{A}$. $\mathcal{B}$ uses this randomness to compute the public keys pk_i for all $i \in [n]$.

Simulating Prep: $\mathcal{B}$ simulates preprocessing honestly for all preprocessing keys in $\mathcal{U}_{\text{Norm}}$, and $\mathcal{U}_{\text{TR}}[j]$ for $j \neq n$, i.e., $\mathcal{B}$ simulates all preproccing keys that do not revoke all parties according to protocol. $\mathcal{B}$'s simulation for keys in $\mathcal{U}_{\text{TR}}[n]$ is more subtle. First, $\mathcal{B}$ picks m random indices $(u_1,\ldots,u_m)$ without replacement from $[d]$. Later, when simulating the challenge ciphertexts, $\mathcal{B}$ will use the preprocessing keys $\{\mathcal{U}_{\text{TR}}[n,u_\gamma]\}_{\gamma\in[m]}$. Now, $\mathcal{B}$ samples $\tilde{z}_i^{(\gamma)} \xleftarrow{\$} \mathbb{Z}_p$ for all $i \in [n]$ and $\gamma \in [m]$. $\mathcal{B}$ simulates $\mathcal{U}_{\text{TR}}[n,u_\gamma]$ such that $\mathcal{U}_{\text{TR}}[n,u_\gamma].r_i \cdot \alpha_i^{(\gamma)} \equiv \tilde{z}_i^{(\gamma)} - s_i$. More formally, for all $\gamma \in [m]$, the reduction simulates $\mathcal{U}_{\text{TR}}[n,u_\gamma]$ as follows:

- Sample $\beta \xleftarrow{\$} \mathbb{Z}_p^*$.
- For all $i \in [n]$, compute $A_i^{(\gamma)} \leftarrow \tilde{z}_i^{(\gamma)}[c_i]_1 - \left[\sum_{\ell\in S} \lambda_{\ell,S}^{(i)} y_\ell c_i\right]_1$. Note that this is equivalent to $[s_i c_i]_1 - \left[\tilde{z}_i^{(\gamma)} c_i\right]_1 \equiv \left[c_i \alpha_i^{(\gamma)} \mathcal{U}_{\text{TR}}[n,u_\gamma].r_i\right]_1$. Then, for all $q \in [m]$ with $q \neq \gamma$, set
$$A_i^{(q)} \leftarrow \frac{\alpha_i^{(q)}}{\alpha_i^{(\gamma)}} \cdot A_i^{(\gamma)} \equiv \frac{\alpha_i^{(q)}}{\alpha_i^{(\gamma)}} \cdot \left[\alpha_i^{(\gamma)} \mathcal{U}_{\text{TR}}[n,u_\gamma].r_i c_i\right]_1 \equiv \left[\alpha_i^{(q)} \mathcal{U}_{\text{TR}}[n,u_\gamma].r_i c_i\right]_1.$$
- For all $i \neq j \in [n]$, compute $B_{i,j}^{(\gamma)} \leftarrow \tilde{z}_j^{(\gamma)} \cdot [c_j/c_i]_1 - \left[\sum_{\ell\in S} \lambda_{\ell,S}^{(j)} y_\ell c_j/c_i\right]_1$. For all $q \in [m]$ with $q \neq \gamma$, compute $B_{i,j}^{(q)} \leftarrow \frac{\alpha_j^{(q)}}{\alpha_j^{(\gamma)}} B_{i,j}^{(\gamma)}$. For all $i \in [n]$ and $q \in [m]$, compute $B_i^{(q)} \leftarrow \sum_{j\in[n]\setminus\{i\}} B_{i,j}^{(q)}$.
- Assemble $\mathcal{U}_{\text{TR}}[n,u_\gamma] \leftarrow \left([\beta]_1, \{A_i^{(q)}, B_i^{(q)}\}_{i\in[n],q\in[m]}\right)$.

For all remaining $\mathcal{U}_{\text{TR}}[n,j]$, where $j \notin \{u_\gamma\}_{\gamma\in[m]}$, simulate preprocessing keys honestly. In the end, $\mathcal{B}$ shuffles all preprocessing keys. Let idx_γ, for all $\gamma \in [m]$, be the index of $\mathcal{U}_{\text{TR}}[n,u_\gamma]$ after shuffling. Denote the resulting preprocessing with $\mathcal{U}$.

Simulating the challenge (k,c). Next, $\mathcal{B}$ simulates the challenge (k_b,c), such that $(k_0,c) \xleftarrow{\$} \mathsf{TrEnc}(\mathsf{tk},n)$, and k_1 is a uniform random key from the keyspace. The goal here is, that k_b is consistent with the internal bit of the (t,n)-Int-EBDDH challenger. To do so, $\mathcal{B}$ sets $k \leftarrow \mathsf{chal}$, and simulates encryption under the preprocessing keys $\{\mathcal{U}_{\text{TR}}[n,u_\gamma]\}_{\gamma\in[m]}$ *without knowledge of the randomness* v from its (t,n)-Int-EBDDH tuple:

$$c \leftarrow \left([v]_2, \left\{ \sum_{i\in[n]} \tilde{z}_i^{(\gamma)} \cdot [vc_i]_1, \mathsf{idx}_\gamma \right\}_{\gamma\in[m]} \right).$$

Note that $\sum_{i\in[n]} \tilde{z}_i^{(\gamma)} \cdot [vc_i]_1 \equiv \sum_{i\in[n]} \left[vc_i(s_i + \mathcal{U}_{\text{TR}}[n,u_\gamma].r_i \cdot \alpha_i^{(\gamma)}) \right]_1$.

Next, $\mathcal{B}$ forwards the preprocessing $\mathcal{U}$, and the challenge (k,c) to $\mathcal{A}$, while simulating the TrEnc oracle as follows.

Simulating the TrEnc oracle. On input $(j \in [0,n]; r_{\mathsf{Enc}})$, $\mathcal{B}$ parses r_{Enc} as $\tilde{v} \in \mathbb{Z}_p$ and r_{Samp}, where r_{Samp} are random coins for sampling from the preprocessing. $\mathcal{B}$ uses r_{Samp} to sample preprocessing keys $(K_1,\mathsf{idx}_1),\ldots,(K_m,\mathsf{idx}_m)$ from $\mathcal{U}_{\text{TR}}[j\ldots n]$ according to the key distribution of $\mathsf{TrEnc}(\mathsf{tk},j)$, which $\mathcal{B}$ can do since it knows the tracing keys. $\mathcal{B}$ computes $(k,c) \leftarrow \mathsf{Enc}(\mathsf{CRS}, \{K_q,\mathsf{idx}_q\}_{q\in[m]}, \tilde{v})$, and outputs (k,c).

Simulating SimPD for the challenge ciphertext. Next, $\mathcal{A}$ asks for simulated partial decryptions on behalf of $\mathcal{I}_{\mathsf{pd}}$. W.l.o.g., assume that $\mathcal{I}_{\mathsf{pd}} \subseteq [n]$ and $|\mathcal{I}_{\mathsf{pd}}| < t$. $\mathcal{B}$ forwards $\mathcal{I}_{\mathsf{pd}}$ to its (t,n)-Int-EBDDH challenger, who replies with

$$\mathsf{view}'_T = \left\{ \left[v \cdot \sum_{\ell\in S} \lambda_{\ell,S}^{(i)} y_\ell \right]_T \right\}_{i\in\mathcal{I}_{\mathsf{pd}}}.$$

Now, observe that, since $s \equiv y_0$ and $s_\ell \equiv y_\ell$ for all $\ell \in [t-1]$, it holds that $s_i \equiv \sum_{\ell\in S} \lambda_{\ell,S}^{(i)} y_\ell$ for all $i \in [n]$, and thus

$$\left[v \cdot \textstyle\sum_{\ell\in S} \lambda_{\ell,S}^{(i)} y_\ell \right]_T \equiv [vs_i]_T \equiv \mathsf{SimPD}(\mathsf{pk},i,c,r_{\mathsf{Enc}}),$$

where r_{Enc} is the randomness used in the challenge ciphertext, i.e., the field element v that is unknown to the reduction, and the randomness used to sample $\{u_\gamma\}_{\gamma\in[m]}$. Hence, $\mathcal{B}$ forwards view'_T to $\mathcal{A}$, who outputs a bit b', which $\mathcal{B}$ forwards to its challenger.

Analysis. Let β denote the hypothetical internal bit of $\mathcal{B}$'s simulation of Game_1 to $\mathcal{A}$, and let b denote the internal bit of the (t,n)-Int-EBDDH challenger. Observe that $\mathcal{B}$ simulates Game_1 perfectly to $\mathcal{A}$ for $\beta = b$. Since $\mathcal{B}$ outputs the same bit as $\mathcal{A}$, it holds that $\mathcal{B}$ wins the Int-EBDDH game whenever $\mathcal{A}$ wins Game_2, and we get that

$$\Pr[\mathsf{Game}_{1,\mathcal{A}}(1^\lambda) = 1] = \Pr[\mathsf{Int\text{-}EBDDH}_{\mathcal{B}}(1^\lambda,n,t) = 1].$$

Applying Claim C.5, we can conclude that for all PPT adversaries $\mathcal{A}$ there exist PPT reductions $\mathcal{B}_1$ and $\mathcal{B}_2$ such that

$$\mathsf{Adv}^{\mathsf{MP}}_{\mathcal{A},\text{Constr. }2}(1^\lambda,t-1) \le 2\cdot \mathsf{Adv}^{\text{hiding}}_{\mathcal{B}_1,\Pi_{\mathsf{Com}}}(1^\lambda) + \mathsf{Adv}^{(t,n)\text{-Int-EBDDH}}_{\mathcal{B}_2,\mathcal{G}}(1^\lambda),$$

which concludes the proof. ∎

D The Game KEYS-IND

In this section, we introduce the game $\delta-$KEYS-IND.

- Global Input: The indices $1,\ldots,Ld$.
- For $U_0,U_1 \subseteq [Ld]$ such that $|U_0| = (L-1)d$ and $|U_1| = d$, we define the following two distributions for sampling without replacement of λ elements:
  - $\mathcal{D}_0$: An element is sampled from U_0 with probability p_α and from U_1 with probability p_β, and is $\perp$ with probability $1-(p_\alpha+p_\beta)$.
  - $\mathcal{D}_1$: An element is sampled from U_0 with probability 0, from U_1 with probability $p_\alpha+p_\beta$ and is $\perp$ with probability $1-(p_\alpha+p_\beta)$.

- The adversary, with access to the the global input and nothing else, returns a decoder $D(q) \to \{0,1\}$ that receives as input a query $q = (k_1, \ldots, k_\lambda)$ where each $k_i \in [Ld] \cup \{\perp\}$.
- The challenger samples U_0, U_1 uniformly at random. Then it applies the following empirical test of the decoder D. For every $b \in \{0,1\}$:
  - For $i \in [Q]$: sample the query q_i according to $\mathcal{D}_b$. Let $a = D(q_i)$. If $a = 1$, then $p_b = p_b + 1$.
- The adversary wins if it holds that $|p_0/Q - p_1/Q| \geq \delta$.

Lemma D.1 *The adversary returns a decoder that does not satisify*

$$|p_1/Q - p_0/Q| \leq \frac{\sqrt{8L}\lambda^{3/2}}{\sqrt{d}} + 2\sqrt{\frac{\lambda}{2Q}}$$

only with negligible probability in λ.

Proof. Let D be a (possibly randomized) decoder with output in $\{0,1\}$. For $b \in \{0,1\}$, let $q = (k_1, \ldots, k_\lambda)$ be drawn from $\mathcal{D}_b$. For a fixed partition $\mathcal{U} = (U_0, U_1)$ (which is hidden from the decoder), we define the advantage as the difference between the probability that the decoder outputs 1 when the query is sampled according to the experiment $b = 0$ and the probability that it outputs 1 when the query is sampled according to the experiment $b = 1$. That is,

$$\mathrm{Adv}(D,\mathcal{U}) := \left| \Pr_{q \leftarrow \mathcal{D}_1^{\mathcal{U}},\, \rho}[D(q;\rho) = 1] - \Pr_{q \leftarrow \mathcal{D}_0^{\mathcal{U}},\, \rho}[D(q;\rho) = 1] \right|,$$

where ρ denotes the internal randomness of the decoder and $\mathcal{D}_b^{\mathcal{U}}$ denotes the distribution $\mathcal{D}_b$ instantiated with the partition $\mathcal{U} = (U_0, U_1)$. Since $D(q;\rho) \in \{0,1\}$, we have

$$\Pr[D(q;\rho) = 1] = \mathbb{E}[D(q;\rho)]\,,$$

and thus

$$\mathrm{Adv}(D,\mathcal{U}) = \left| \mathbb{E}_{q \leftarrow \mathcal{D}_1^{\mathcal{U}},\, \rho}[D(q;\rho)] - \mathbb{E}_{q \leftarrow \mathcal{D}_0^{\mathcal{U}},\, \rho}[D(q;\rho)] \right|.$$

We prove the following claim in Section D.1.

Claim D.2 For every D and uniformly distributed $\mathcal{U}$ it holds that

$$\mathrm{Adv}(D,\mathcal{U}) \leq \frac{\sqrt{8L}\lambda^{3/2}}{\sqrt{d}}$$

except with negligible probability in λ.

Equipped with Claim D.2, we can next bound $|p_0/Q - p_1/Q|$. First, note that p_b/Q is an estimation of the expectation $\mu_b = \mathbb{E}_{q \leftarrow \mathcal{D}_b^{\mathcal{U}},\, \rho}[D(q;\rho)]$. Hence, by Hoeffding's inequality, it holds that

$$\Pr\left[\left|\frac{p_b}{Q} - \mu_b\right| \geq \epsilon\right] \leq 2\exp(-2Q\epsilon^2).$$

In particular, if $\epsilon \geq \sqrt{\lambda/(2Q)}$, then

$$\left|\frac{p_b}{Q} - \mu_b\right| \leq \epsilon$$

except with negligible probability in λ.

Next, observe that

$$\frac{p_1}{Q} - \frac{p_0}{Q} = \left(\frac{p_1}{Q} - \mu_1\right) + (\mu_1 - \mu_0) - \left(\frac{p_0}{Q} - \mu_0\right).$$

Therefore, by the triangle inequality,

$$\left|\frac{p_1}{Q} - \frac{p_0}{Q}\right| \leq \left|\frac{p_1}{Q} - \mu_1\right| + |\mu_1 - \mu_0| + \left|\frac{p_0}{Q} - \mu_0\right|.$$

If

$$\left|\frac{p_b}{Q} - \mu_b\right| \leq \epsilon \qquad \text{for } b \in \{0,1\},$$

then

$$\left|\frac{p_1}{Q} - \frac{p_0}{Q}\right| \leq |\mu_1 - \mu_0| + 2\epsilon = \mathsf{Adv}(D,\mathcal{U}) + 2\epsilon.$$

In particular, for $\epsilon = \sqrt{\lambda/(2Q)}$,

$$\left|\frac{p_1}{Q} - \frac{p_0}{Q}\right| \leq \mathsf{Adv}(D,\mathcal{U}) + 2\sqrt{\frac{\lambda}{2Q}}.$$

Then, the proof is concluded by applying Claim D.2. ∎

D.1 Proof of Claim D.2

Proof. Expanding the expectation over q, we obtain

$$\mathrm{Adv}(D,\mathcal{U}) = \left|\sum_{q \leftarrow \mathcal{D}_1^{\mathcal{U}}} \Pr[q] \cdot \mathbb{E}_\rho[D(q;\rho)] - \sum_{q \leftarrow \mathcal{D}_0^{\mathcal{U}}} \Pr[q] \cdot \mathbb{E}_\rho[D(q;\rho)]\right|.$$

Let us fix any D. Next, for each partition $\mathcal{U}$, define

$$f(\mathcal{U}) := \sum_{q \leftarrow \mathcal{D}_1^{\mathcal{U}}} \Pr[q] \cdot \mathbb{E}_\rho[D(q;\rho)] - \sum_{q \leftarrow \mathcal{D}_0^{\mathcal{U}}} \Pr[q] \cdot \mathbb{E}_\rho[D(q;\rho)]\,.$$

We consider the expectation $\mathbb{E}_{\mathcal{U}}[f(\mathcal{U})]$, where the average is over the choice of the partition $\mathcal{U}$.

Intutively, it holds that $\mathbb{E}_{\mathcal{U}}[f(\mathcal{U})] = 0$. Indeed, given a query q, the decoder can not distinguish if it comes from sampling according to $\mathcal{D}_0$ or $\mathcal{D}_1$. This is true since the sampling is without replacement, so the decoder cannot learn any information about the size of the set given the query. We prove this formally in Claim D.3.

Claim D.3 $\mathbb{E}_{\mathcal{U}}[f(\mathcal{U})] = 0$.

We defer the proof to Appendix D.2. We next show how to conclude the proof from this claim. Now we can bound $|f(\mathcal{U})| = \mathsf{Adv}(D,\mathcal{U})$ using concentration around the mean. In particular, for every $\epsilon > 0$,

$$\Pr_{\mathcal{U}}[\mathsf{Adv}(D,\mathcal{U}) \geq \epsilon] = \Pr_{\mathcal{U}}[|f(\mathcal{U}) - \mathbb{E}_{\mathcal{U}}[f(\mathcal{U})]| \geq \epsilon]\,.$$

Let $\mathcal{U} = (U_0, U_1)$. We next bound the change in f when exchanging one element of U_0 with one element of U_1. Let $x \in U_0$ and $y \in U_1$, and define

$$U_0' = (U_0 \setminus \{x\}) \cup \{y\}, \qquad U_1' = (U_1 \setminus \{y\}) \cup \{x\}.$$

Let τ denote the permutation of $[Ld]$ that swaps x and y and fixes every other element. Furthermore, write

$$g(q) := \mathbb{E}_\rho[D(q;\rho)] \in [0,1].$$

For each $b \in \{0,1\}$, if $q \leftarrow \mathcal{D}_b^{\mathcal{U}}$, then $\tau(q)$ is distributed exactly according to $\mathcal{D}_b^{\mathcal{U}'}$. Thus, we have

$$\mathbb{E}_{q' \leftarrow \mathcal{D}_b^{\mathcal{U}'}}[g(q')] = \mathbb{E}_{q \leftarrow \mathcal{D}_b^{\mathcal{U}}}[g(\tau(q))].$$

Hence,

$$\left|\mathbb{E}_{q \leftarrow \mathcal{D}_b^{U}}[g(q)] - \mathbb{E}_{q' \leftarrow \mathcal{D}_b^{\mathcal{U}'}}[g(q')]\right| = \left|\mathbb{E}_{q \leftarrow \mathcal{D}_b^{\mathcal{U}}}[g(q) - g(\tau(q))]\right|.$$

Since $g(q) \in [0,1]$, and $g(q) = g(\tau(q))$ whenever q contains neither x nor y, we have, for every $b \in \{0,1\}$,

$$\mathbb{E}_{q \leftarrow \mathcal{D}_b^{\mathcal{U}}}[|g(q) - g(\tau(q))|] \leq \Pr_{q \leftarrow \mathcal{D}_b^{U}}[x \in q] + \Pr_{q \leftarrow \mathcal{D}_b^{\mathcal{U}}}[y \in q],$$

where the last inequality follows from the union bound, since we only need to account for the events in which $x \in q$ or $y \in q$.

For $b = 1$, we have

$$\Pr_{q \leftarrow \mathcal{D}_1^{\mathcal{U}}}[x \in q] = 0, \qquad \Pr_{q \leftarrow \mathcal{D}_1^{\mathcal{U}}}[y \in q] = \frac{\lambda(p_\alpha + p_\beta)}{d}.$$

For $b = 0$, we have

$$\Pr_{q \leftarrow \mathcal{D}_0^{\mathcal{U}}}[x \in q] = \frac{\lambda p_\alpha}{(L-1)d}, \qquad \Pr_{q \leftarrow \mathcal{D}_0^{\mathcal{U}}}[y \in q] = \frac{\lambda p_\beta}{d},$$

and therefore, since $L \geq 2$,

$$\frac{\lambda p_\alpha}{(L-1)d} + \frac{\lambda p_\beta}{d} \leq \frac{\lambda(p_\alpha + p_\beta)}{d}.$$

Thus, for both $b \in \{0, 1\}$,

$$\mathbb{E}_{q \leftarrow \mathcal{D}_b^{\mathcal{U}}}\left[|g(q) - g(\tau(q))|\right] \leq \frac{\lambda(p_\alpha + p_\beta)}{d}.$$

Consequently, for both $b \in \{0, 1\}$,

$$\begin{aligned}\left|\mathbb{E}_{q \leftarrow \mathcal{D}_b^{\mathcal{U}}}[g(q)] - \mathbb{E}_{q \leftarrow \mathcal{D}_b^{\mathcal{U}'}}[g(q)]\right| &= \left|\mathbb{E}_{q \leftarrow \mathcal{D}_b^{\mathcal{U}}}\left[g(q) - g(\tau(q))\right]\right| \\ &\leq \mathbb{E}_{q \leftarrow \mathcal{D}_b^{\mathcal{U}}}\left[|g(q) - g(\tau(q))|\right] \\ &\leq \frac{\lambda(p_\alpha + p_\beta)}{d}.\end{aligned}$$

It follows that

$$\begin{aligned}|f(\mathcal{U}) - f(\mathcal{U}')| &\leq \left|\mathbb{E}_{q \leftarrow \mathcal{D}_1^{\mathcal{U}}}[g(q)] - \mathbb{E}_{q \leftarrow \mathcal{D}_1^{\mathcal{U}'}}[g(q)]\right| \\ &\qquad + \left|\mathbb{E}_{q \leftarrow \mathcal{D}_0^{\mathcal{U}}}[g(q)] - \mathbb{E}_{q \leftarrow \mathcal{D}_0^{\mathcal{U}'}}[g(q)]\right| \\ &\leq \frac{2\lambda(p_\alpha + p_\beta)}{d} \\ &\leq \frac{2\lambda}{d},\end{aligned}$$

where the last inequality follows from $p_\alpha + p_\beta \leq 1$.

We now represent a uniformly random partition $\mathcal{U}$ by a uniformly random permutation $\pi = (\pi_1, \ldots, \pi_{Ld})$ of $[Ld]$. Namely, define

$$U_0(\pi) = \{\pi_1, \ldots, \pi_{(L-1)d}\}, \qquad U_1(\pi) = \{\pi_{(L-1)d+1}, \ldots, \pi_{Ld}\},$$

and let

$$F(\pi) := f\big(U_0(\pi), U_1(\pi)\big).$$

A uniformly random permutation induces a uniformly random partition with $|U_0| = (L-1)d$ and $|U_1| = d$. Consider now a transposition of two entries of π, as defined in Lemma A.5. If both entries lie in the same block, then the induced partition does not change, and hence F does not change. If the entries lie in different blocks, the transposition exchanges one element of U_0 with one element of U_1. By the bound above,

$$|F(\pi) - F(\pi')| \leq \frac{2\lambda}{d}.$$

Therefore, Lemma A.5 applies with

$$c_i = \frac{2\lambda}{d}, \qquad i = 1, \ldots, Ld.$$

Consequently,

$$\begin{aligned}\Pr_{\mathcal{U}}[|f(\mathcal{U}) - \mathbb{E}_{\mathcal{U}}[f(\mathcal{U})]| \geq \epsilon] &\leq 2\exp\left(-\frac{\epsilon^2}{2\sum_{i=1}^{Ld} c_i^2}\right) \\ &= 2\exp\left(-\frac{\epsilon^2}{2Ld(2\lambda/d)^2}\right) \\ &= 2\exp\left(-\frac{d\epsilon^2}{8L\lambda^2}\right).\end{aligned}$$

Since by Claim D.3, $\mathbb{E}_{\mathcal{U}}[f(\mathcal{U})] = 0$ and $\mathsf{Adv}(D,\mathcal{U}) = |f(\mathcal{U})|$, we obtain

$$\Pr_{\mathcal{U}}[\mathsf{Adv}(D,\mathcal{U}) \geq \epsilon] \leq 2\exp\left(-\frac{d\epsilon^2}{8L\lambda^2}\right).$$

Finally, setting

$$\epsilon = \frac{\sqrt{8L}\,\lambda^{3/2}}{\sqrt{d}}$$

gives

$$\Pr_{\mathcal{U}}\left[\mathsf{Adv}(D,\mathcal{U}) \geq \frac{\sqrt{8L}\,\lambda^{3/2}}{\sqrt{d}}\right] \leq 2e^{-\lambda},$$

which is negligible in λ. This proves the claim.

D.2 The proof of Claim D.3

Note that if it holds that

$$\mathbb{E}_{\mathcal{U}}[\Pr[q \mid \mathcal{D}_1^{\mathcal{U}}]] = \mathbb{E}_{\mathcal{U}}[\Pr[q \mid \mathcal{D}_0^{\mathcal{U}}]] \quad \text{for all } q, \tag{4}$$

where the expectation is over a uniformly random partition $\mathcal{U} = (U_0, U_1)$, and $\Pr[q \mid \mathcal{D}_b^{\mathcal{U}}]$ denotes the probability that $\mathcal{D}_b^{\mathcal{U}}$ outputs the query q, then by linearity of expectation it follows that $\mathbb{E}_{\mathcal{U}}[f(\mathcal{U})] = 0$. To see this, observe that since the decoder does not depend on $\mathcal{U}$ (as it is chosen after D was constructued), we can write

$$f(\mathcal{U}) = \sum_q \Pr[q \mid \mathcal{D}_1^{\mathcal{U}}]\, g(q) - \sum_q \Pr[q \mid \mathcal{D}_0^{\mathcal{U}}]\, g(q),$$

where $g(q) := \mathbb{E}_\rho[D(q;\rho)]$. Taking expectation over $\mathcal{U}$ and using linearity, we obtain

$$\mathbb{E}_{\mathcal{U}}[f(\mathcal{U})] = \sum_q g(q)\, \mathbb{E}_{\mathcal{U}}[\Pr[q \mid \mathcal{D}_1^{\mathcal{U}}]] - \sum_q g(q)\, \mathbb{E}_{\mathcal{U}}[\Pr[q \mid \mathcal{D}_0^{\mathcal{U}}]].$$

The claim follows since, by Equation 4, the two terms are equal for every q.

In order to prove Equation 4, observe that for $b \in \{0,1\}$ and every q it holds that:

$$\mathbb{E}_{\mathcal{U}}\left[\Pr[q \mid \mathcal{D}_b^{\mathcal{U}}]\right] = \sum_{\mathcal{U}} \Pr[\mathcal{U}] \Pr_{q' \leftarrow \mathcal{D}_b^{\mathcal{U}}}[q' = q] = \Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_b^{\mathcal{U}}}[q' = q].$$

Thus, it suffices to compute the probability that a fixed query q is output by the above experiment under $b = 0$ and $b = 1$, and show that they are the same.

Observe that under both distributions $D_0^{\mathcal{U}}$ and $D_1^{\mathcal{U}}$, the total probability of sampling a non-$\perp$ element is $p_\alpha + p_\beta$. Hence, under both distributions, the number c of non-$\perp$ elements in the query has the same distribution. Since queries are considered unordered, conditioned on c the $\perp$ entries carry no further information. We may therefore fix c and identify a query with the set of its c non-$\perp$ elements.

We define the following set for every $c, a \in [\lambda]$ where $1 \leq a \leq c$:

$$Q^c_{a,\mathcal{K}} = \{q \subseteq U_0 \cup U_1 \mid |q| = c, |q \cap U_0| = a, |q \cap U_1| = c - a\}.$$

Computing $\Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_1^{\mathcal{U}}}[q' = q]$. We compute the probability that a fixed query q is output when the partition $\mathcal{U}$ is sampled uniformly at random and then q' is drawn from $\mathcal{D}_1^{\mathcal{U}}$. Observe that the only queries with c keys from $U_0 \cup U_1$ that can be sampled according to $\mathcal{D}_1^{\mathcal{U}}$ are those contained entirely in U_1, namely the queries in $Q^c_{0,\mathcal{U}}$. Since $|U_1| = d$, it holds that $|Q^c_{0,\mathcal{U}}| = \binom{d}{c}$. According to the sampling rule, the probability that the query contains c keys from U_1 is $\binom{\lambda}{c}(p_\alpha + p_\beta)^c(1 - p_\alpha - p_\beta)^{\lambda - c}$. Hence, each query in $Q^c_{0,\mathcal{U}}$ is sampled with probability

$$\frac{\binom{\lambda}{c}(p_\alpha + p_\beta)^c(1 - p_\alpha - p_\beta)^{\lambda - c}}{\binom{d}{c}}.$$

Now fix a query q with exactly c non-$\perp$ values. Such a query can be sampled from $\mathcal{D}_1^{\mathcal{K}}$ only if $q \subseteq U_1$. Since U_1 is a uniformly random subset of size d of $[Ld]$, we have

$$\Pr[q \subseteq U_1] = \frac{\binom{Ld-c}{d-c}}{\binom{Ld}{d}},$$

where the probability is over the choice of U_1.

Note that it holds that equals

$$\Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_1^{\mathcal{U}}}[q' = q] = \Pr[q \subseteq U_1] \cdot \Pr[q \text{ is sampled} \mid q \subseteq U_1].$$

Thus,

$$\Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_1^{\mathcal{U}}}[q' = q] = \frac{\binom{Ld-c}{d-c}}{\binom{Ld}{d}} \cdot \frac{\binom{\lambda}{c}(p_\alpha + p_\beta)^c(1-p_\alpha-p_\beta)^{\lambda-c}}{\binom{d}{c}}.$$

Using the identity

$$\binom{Ld}{d}\binom{d}{c} = \binom{Ld}{c}\binom{Ld-c}{d-c},$$

we obtain

$$\Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_1^{\mathcal{U}}}[q' = q] = \frac{\binom{\lambda}{c}(p_\alpha + p_\beta)^c(1-p_\alpha-p_\beta)^{\lambda-c}}{\binom{Ld}{c}}.$$

Computing $\Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_0^{\mathcal{U}}}[q' = q]$. We compute the probability that a fixed query q is output when the partition $\mathcal{U}$ is sampled uniformly at random and then q' is drawn from $\mathcal{D}_0^{\mathcal{U}}$. First note that since $|U_0| = (L-1)d$ and $|U_1| = d$, we have

$$|Q^c_{a,\mathcal{U}}| = \binom{(L-1)d}{a}\binom{d}{c-a}.$$

According to the sampling rule, the probability that the query contains exactly a keys from U_0 and $c-a$ keys from U_1 is

$$\binom{\lambda}{a}\binom{\lambda-a}{c-a}p_\alpha^a p_\beta^{c-a}(1-p_\alpha-p_\beta)^{\lambda-c}.$$

Hence, each query in $Q^c_{a,\mathcal{U}}$ is sampled with probability

$$p_1(a,c) = \frac{\binom{\lambda}{a}\binom{\lambda-a}{c-a}p_\alpha^a p_\beta^{c-a}(1-p_\alpha-p_\beta)^{\lambda-c}}{\binom{(L-1)d}{a}\binom{d}{c-a}}.$$

Now fix a query q of exactly c non-$\perp$ values. Such a query is sampled from $\mathcal{D}_0$ if it contains a elements from U_0 and $c-a$ elements from U_1 for some $a \in \{0,\ldots,c\}$. Thus,

$$\Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_0^{\mathcal{U}}}[q' = q] = \sum_{a=0}^{c} \Pr\big[|q \cap U_0| = a \text{ and } |q \cap U_1| = c-a\big] \cdot p_1(a,c).$$

Since U_1 is a uniformly random subset of size d, we have

$$\Pr\big[|q \cap U_0| = a \text{ and } |q \cap U_1| = c-a\big] = \frac{\binom{c}{c-a}\binom{Ld-c}{d-c+a}}{\binom{Ld}{d}}.$$

Therefore,

$$\Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_0^{\mathcal{U}}}[q' = q] = \sum_{a=0}^{c} \frac{\binom{c}{c-a}\binom{Ld-c}{d-c+a}}{\binom{Ld}{d}} \cdot \frac{\binom{\lambda}{a}\binom{\lambda-a}{c-a}p_\alpha^a p_\beta^{c-a}(1-p_\alpha-p_\beta)^{\lambda-c}}{\binom{(L-1)d}{a}\binom{d}{c-a}}.$$

Using the identity

$$\frac{\binom{c}{c-a}\binom{Ld-c}{d-c+a}}{\binom{Ld}{d}} = \frac{\binom{(L-1)d}{a}\binom{d}{c-a}}{\binom{Ld}{c}},$$

we obtain

$$\Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_0^{\mathcal{U}}}[q' = q] = \frac{1}{\binom{Ld}{c}} \sum_{a=0}^{c} \binom{\lambda}{a}\binom{\lambda-a}{c-a}p_\alpha^a p_\beta^{c-a}(1-p_\alpha-p_\beta)^{\lambda-c}.$$

Using

$$\binom{\lambda}{a}\binom{\lambda-a}{c-a} = \binom{\lambda}{c}\binom{c}{a},$$

we obtain

$$\begin{aligned}\Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_0^{\mathcal{U}}}[q' = q] &= \frac{1}{\binom{Ld}{c}} \sum_{a=0}^{c} \binom{\lambda}{a} \binom{\lambda - a}{c - a} p_\alpha^a \, p_\beta^{c-a} \, (1 - p_\alpha - p_\beta)^{\lambda - c} \\ &= \frac{1}{\binom{Ld}{c}} \binom{\lambda}{c} (1 - p_\alpha - p_\beta)^{\lambda - c} \sum_{a=0}^{c} \binom{c}{a} p_\alpha^a \, p_\beta^{c-a} \\ &= \frac{1}{\binom{Ld}{c}} \binom{\lambda}{c} (1 - p_\alpha - p_\beta)^{\lambda - c} (p_\alpha + p_\beta)^c,\end{aligned}$$

where the last equality follows from the binomial theorem.
We conclude that

$$\Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_0^{\mathcal{U}}}[q' = q] = \frac{\binom{\lambda}{c}(p_\alpha + p_\beta)^c (1 - p_\alpha - p_\beta)^{\lambda - c}}{\binom{Ld}{c}} = \Pr_{\mathcal{U},\, q' \leftarrow \mathcal{D}_1^{\mathcal{U}}}[q' = q].$$

According to the discussion above, we conclude that $\mathbb{E}_{\mathcal{U}}[f(\mathcal{U})] = 0$. ∎

E A Proof of Assumption 3

In this section, we prove that Assumption 3 holds in the generic group model [47, 46], making use of the following master theorem.

Theorem E.1 (Master Theorem [6, 21]). *Let $p \in \mathbb{N}$ be a prime, let $n, v_1, v_2, v_t \in \mathbb{N}$, let $\mathbf{L}_1 \in \mathbb{F}_p[X_1, \ldots, X_n]^{v_1}$, $\mathbf{L}_2 \in \mathbb{F}_p[X_1, \ldots, X_n]^{v_2}$, $\mathbf{L}_T \in \mathbb{F}_p[X_1, \ldots X_n]^{v_t}$, and $f \in \mathbb{F}_p[X_1, \ldots, X_n]$ be tuples of n-variate polynomials of maximum degree $d_{\mathbf{L}_1}, d_{\mathbf{L}_2}$, $d_{\mathbf{L}_T}$, and d_f respectively. Denote $v = v_1 + v_2 + v_t$ and $d = \max(d_{\mathbf{L}_1} + d_{\mathbf{L}_2}, d_{\mathbf{L}_T}, d_f)$. If f is independent of $(\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_T)$, then for any generic adversary $\mathcal{A}$ that makes at most q group oracle queries:*

$$\left| \Pr\left[\mathcal{A}\begin{pmatrix} p, \mathsf{h}_1[\mathbf{L}_1(x)], \mathsf{h}_2[\mathbf{L}_2(x)], \\ \mathsf{h}_T[\mathbf{L}_T(x), f(x)] \end{pmatrix} = 1\right] - \Pr\left[\mathcal{A}\begin{pmatrix} p, \mathsf{h}_1[\mathbf{L}_1(x)], \mathsf{h}_2[\mathbf{L}_2(x)], \\ \mathsf{h}_T[\mathbf{L}_T(x), r] \end{pmatrix} = 1\right] \right| \leq \frac{(q + v + 2)^2 \cdot d}{2p},$$

where $\mathsf{h}_1, \mathsf{h}_2$, and h_T denote the corresponding handles, and the probabilities are taken over the choice of $x \xleftarrow{\$} \mathbb{F}_p^n$ and $r \xleftarrow{\$} \mathbb{F}_p$.

Definition E.2 (Independence). Let the completion $\mathbf{C}(\mathbf{L}) = \{\mathbf{L}_1 \otimes \mathbf{L}_2\} \cup \mathbf{L}_T$ of size D for lists $(\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_T)$ be of the form

$$\mathbf{C}(\mathbf{L}) = \{g_1(X_1, \ldots, X_n), \ldots g_D(X_1, \ldots, X_n)\}.$$

We say that an element f is dependent on lists $(\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_T)$, if there exist coefficients $\kappa_i \in \mathbb{F}_p$ such that

$$f(X_1, \ldots, X_n) = \sum_{i=1}^{D} \kappa_i \cdot g_i(X_1, \ldots, X_n).$$

Otherwise, we say that f is independent of $(\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_T)$.

Lemma E.3 *The (t, n)-Int-EBDDH assumption (Assumption 3) holds in the generic group model.*

The proof of Lemma E.3 follows a pattern similar to that of [18, Lemma E.3]. In the following proof, we assume that, when generating the pairing ensemble $E \xleftarrow{\$} \mathcal{G}(\lambda)$, the generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ are sampled at random.

Proof. Ideally, we would like to apply the master theorem directly to prove that (t, n)-Int-EBDDH holds in the GGM by showing that the challenge $y_0 v$ is symbolically not computable from the elements in the adversary's view. However, some of the elements in the view are fractions, e.g., $\{[1/c_i]_1\}_{i \in [n]}$, whereas the master theorem is stated for polynomials only.

To deal with this, we first apply two game hops to the (t, n)-Int-EBDDH assumption. Let Game_0 be identical to the (t, n)-Int-EBDDH experiment (Figure 10). W.l.o.g., we rewrite the random generators $[1]_1$ and $[1]_2$ as $[w_1]_1$ and $[w_2]_2$ for independent random $w_1, w_2 \xleftarrow{\$} \mathbb{Z}_p^*$.
$\underline{\mathsf{Game}_1}$: In Game_1 the challenger initially samples independent random $\widetilde{w_1}, \widetilde{w_2} \xleftarrow{\$} \mathbb{Z}_p^*$ and sets the generators $[w_1]_1 \leftarrow \left[\widetilde{w_1} \cdot \prod_{i \in [n]} c_i\right]_1$ and $[w_2]_2 \leftarrow \left[\widetilde{w_2} \cdot \prod_{i \in [n]} c_i\right]_2$. The challenger then continues to simulate the game with respect to these generators.

Claim E.4 For all PPT adversaries $\mathcal{A}$, it holds that

$$\Pr\big[\mathsf{Game}_{1,\mathcal{A}}(1^\lambda, n, t)\big] = \Pr\big[\mathsf{Game}_{0,\mathcal{A}}(1^\lambda, n, t)\big].$$

Proof of Claim E.4. Since $c_i \in \mathbb{Z}_p^*$ and thus invertible for all $i \in [n]$, the distribution of the generators in Game_1 remains uniform and independent. ∎

$\underline{\mathsf{Game}_2}$: In Game_2, we instead sample $c_i \xleftarrow{\$} \mathbb{Z}_p$ for all $i \in [n]$ and $\widetilde{w_1}, \widetilde{w_2} \xleftarrow{\$} \mathbb{Z}_p$ (instead of sampling from $\mathbb{Z}_p^*$).

Claim E.5 For all PPT adversaries $\mathcal{A}$, it holds that

$$\big|\Pr\big[\mathsf{Game}_{2,\mathcal{A}}(1^\lambda, n, t) = 1\big] - \Pr\big[\mathsf{Game}_{1,\mathcal{A}}(1^\lambda, n, t) = 1\big]\big| \leq \frac{n+2}{p}.$$

Proof of Claim E.5. Clearly, both games are identical unless any of the variables $\{c_i\}_{i\in[n]}$, $\widetilde{w_1}$, or $\widetilde{w_2}$ is 0, which, by a union bound, happens with probability $(n+2)/p$ in Game_2. ∎

It remains to bound $\mathcal{A}$'s advantage in Game_2. In Game_2, every field element sampled by the challenger is uniform over $\mathbb{Z}_p$, matching the hypothesis of the master theorem. Let $S = \{0, \ldots, t-1\}$. We introduce the formal variables $\widetilde{W_1}, \widetilde{W_2}, \{Y_\ell\}_{\ell\in S}, \{C_i\}_{i\in[n]}$, and V corresponding to the field elements $\widetilde{w_1}$, $\widetilde{w_2}$, $\{y_\ell\}_{\ell\in S}$, $\{c_i\}_{i\in[n]}$, and v sampled by the challenger in Game_2. Instead of receiving group elements directly, the adversary receives corresponding handles and can perform generic operations on the handles using an addition and a pairing oracle. W.l.o.g., assume that the adversary, after receiving handles to $\mathsf{view}_1, \mathsf{view}_2$, and view_T as well as the challenge chal and the group elements in E, outputs a set $\mathcal{I} \subseteq [n]$ with $|\mathcal{I}| < t$ (in particular, $0 \notin \mathcal{I}$). As a response, the adversary receives a set of handles to the elements in view'_T.

Consider the following lists of rational functions in our formal variables:

$$\mathbf{L}'_1 = \left(1, \{C_i\}_{i\in[n]}, \{VC_i\}_{i\in[n]}, \left\{\sum_{\ell\in S} \lambda_{\ell,S}^{(i)} Y_\ell C_i\right\}_{i\in[n]}, \{C_i/C_j\}_{i\neq j\in[n]}, \left\{\sum_{\ell\in S} \lambda_{\ell,S}^{(i)} Y_\ell C_i/C_j\right\}_{i\neq j\in[n]}\right)$$
$$\mathbf{L}'_2 = \left(1, V, \{1/C_i\}_{i\in[n]}\right)$$
$$\mathbf{L}'_T = \left(Y_0, \left\{V \cdot \sum_{\ell\in S} \lambda_{\ell,S}^{(i)} Y_\ell\right\}_{i\in\mathcal{I}}\right)$$

alongside $f' = Y_0 V$.

Also, let $W_1 = \widetilde{W_1} \cdot \prod_{i\in[n]} C_i$ and $W_2 = \widetilde{W_2} \cdot \prod_{i\in[n]} C_i$. Then, we can express the adversary's view in Game_2 as lists of polynomials in our formal variables, where $\mathbf{L}_1$, $\mathbf{L}_2$, and $\mathbf{L}_T$ correspond to elements from $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$, respectively. In the following, multiplication with lists applies component-wise.

$$\begin{aligned}
\mathbf{L}_1 &= W_1 \cdot \mathbf{L}'_1, \\
\mathbf{L}_2 &= W_2 \cdot \mathbf{L}'_2, \\
\mathbf{L}_T &= W_1 W_2 \cdot \mathbf{L}'_T, \text{ and the challenge} \\
f &= W_1 W_2 f'
\end{aligned}$$

Let $\mathbf{C} = \mathbf{C}(\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_T) = \{\mathbf{L}_1 \otimes \mathbf{L}_2\} \cup \mathbf{L}_T$ and $\mathbf{C}' = \mathbf{C}(\mathbf{L}'_1, \mathbf{L}'_2, \mathbf{L}'_T) = \{\mathbf{L}'_1 \otimes \mathbf{L}'_2\} \cup \mathbf{L}'_T$ denote the completions, respectively. Since $\mathbf{L}'_1, \mathbf{L}'_2$ contain fractions, entries of $\mathbf{C}'$ are not exclusively polynomials: We regard $\mathbf{C}'$ and f' as (a list of) elements of the field of rational functions $\mathbb{F} := \mathbb{F}_p(Y_0, \ldots, Y_{t-1}, C_1, \ldots, C_n, V, \widetilde{W_1}, \widetilde{W_2})$, and understand dependence for f' and $\mathbf{C}'$ in the sense of Definition E.2 extended to $\mathbb{F}$ (i.e., we allow the κ_i-combination to range over elements of $\mathbb{F}$ rather than only polynomials).

Claim E.6 f is independent of $\mathbf{C}$ if and only if f' is independent of $\mathbf{C}'$.

Proof of Claim E.6. Let $g'_1, \ldots, g'_D$ be the entries of $\mathbf{C}'$. It holds by construction that $\mathbf{C} = W_1 W_2 \mathbf{C}'$ entry-wise. Hence, f is dependent on $\mathbf{C}$ if and only if there exists $\kappa_1, \ldots, \kappa_D \in \mathbb{Z}_p$ such that

$$W_1 W_2 \cdot f' = f = \sum_{i=1}^{D} \kappa_i (W_1 W_2 g'_i) = W_1 W_2 \cdot \sum_{i=1}^{D} \kappa_i g'_i.$$

Since $W_1 W_2 = \widetilde{W_1}\widetilde{W_2} \prod_{i\in[n]} C_i^2$ is nonzero over the field $\mathbb{F}$, we can cancel $W_1 W_2$ on both sides, and thus f is dependent of $\mathbf{C}$ iff f' is dependent on $\mathbf{C}'$. The claim follows by contraposition. ∎

Claim E.7 Every entry in $\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_T$, and f is a polynomial $\mathbb{F}_p[Y_0, \ldots, Y_{t-1}, C_1, \ldots, C_n, V, \widetilde{W_1}, \widetilde{W_2}]$ of degree at most $n+3$, $n+2$, for $\mathbf{L}_1$ and $\mathbf{L}_2$ and $2n+4$ for $\mathbf{L}_T$ and f, respectively.

Proof of Claim E.7. Since W_1 and W_2 each contain every C_i as a factor, multiplying by $W_1 W_2$ cancels every denominator. The stated degree bounds follow by inspection. ∎

Claim E.8 f' is independent of $\mathbf{C}'$.

Proof of Claim E.8. Let us first inspect the completion.

$$\begin{aligned}
\mathbf{C}' = \Bigg\{ & 1, \{C_i\}_{i\in[n]}, \{VC_i\}_{i\in[n]}, \left\{\sum_{\ell\in S} \lambda_{\ell,S}^{(i)} Y_\ell C_i\right\}_{i\in[n]}, \{C_i/C_j\}_{i\neq j\in[n]}, \left\{\sum_{\ell\in S} \lambda_{\ell,S}^{(i)} Y_\ell C_i/C_j\right\}_{i\neq j\in[n]}, \\
& V, \{1/C_i\}_{i\in[n]}, Y_0, \left\{V \cdot \sum_{\ell\in S} \lambda_{\ell,S}^{(i)} Y_\ell\right\}_{i\in\mathcal{I}}, \{V^2 C_i\}_{i\in[n]}, \left\{V \cdot \sum_{\ell\in S} \lambda_{\ell,S}^{(i)} Y_\ell C_i\right\}_{i\in[n]}, \{VC_i/C_j\}_{i\neq j\in[n]}, \\
& \left\{V \cdot \sum_{\ell\in S} \lambda_{\ell,S}^{(i)} Y_\ell C_i/C_j\right\}_{i\neq j\in[n]}, \left\{\sum_{\ell\in S} \lambda_{\ell,S}^{(i)} Y_\ell\right\}_{i\in[n]}, \{C_i/C_j C_{j'}\}_{i,j,j'\in[n], i\neq j}, \\
& \left\{\sum_{\ell\in S} \lambda_{\ell,S}^{(i)} Y_\ell C_i/C_j C_{j'}\right\}_{i,j,j'\in[n], i\neq j} \Bigg\}
\end{aligned}$$

To prove independence, we have to show that f' cannot be represented as a linear combination of elements $\mathbf{C}'$. Observe that the only elements that contain f' as a monomial are $\left\{V \cdot \sum_{\ell\in S} \lambda_{\ell,S}^{(i)} Y_\ell\right\}_{i\in\mathcal{I}}$. Moreover, none of the remaining terms contain $x \cdot VY_\ell$ as a monomial for any $\ell \in S$, and any $\kappa \in \mathbb{Z}_p^*$. As a result, none of the remaining terms can contribute to a linear combination of f' in a non-trivial manner. As a result, we get that f' is independent of the completion, unless there exists $\kappa_1, \ldots, \kappa_{t-1} \in \mathbb{Z}_p$ such that

$$f' = VY_0 = \sum_{i=1}^{t-1} \kappa_i V \sum_{\ell\in S} \lambda_{\ell,S}^{(p_i)} Y_\ell,$$

where $\{p_i \mid i \in [t-1]\} = \mathcal{I}$. We can divide V on both sides and factor out the various Y_ℓ to get

$$Y_0 = \left(Y_0 \cdot \sum_{i=1}^{t-1} \kappa_i \lambda_{0,S}^{(p_i)}\right) + \sum_{\ell=1}^{t-1} Y_\ell \sum_{i=1}^{t-1} \kappa_i \lambda_{\ell,S}^{(p_i)}.$$

Note that this is equivalent to the following equation system

$$\underbrace{\begin{pmatrix} \lambda_{0,S}^{(p_1)} & \lambda_{0,S}^{(p_2)} & \cdots & \lambda_{0,S}^{(p_{t-1})} \\ \lambda_{1,S}^{(p_1)} & \lambda_{1,S}^{(p_2)} & \cdots & \lambda_{1,S}^{(p_{t-1})} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{t-1,S}^{(p_1)} & \lambda_{t-1,S}^{(p_2)} & \cdots & \lambda_{t-1,S}^{(p_{t-1})} \end{pmatrix}}_{M} \cdot \underbrace{\begin{pmatrix} \kappa_1 \\ \kappa_2 \\ \vdots \\ \kappa_{t-1} \end{pmatrix}}_{\kappa} = \underbrace{\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}}_{\mathbf{b}},$$

where $M \in \mathbb{Z}_p^{t\times(t-1)}$, and $\lambda_{i,S}^{(p)}$ is the Lagrange coefficient for index i to interpolate among set S at position p. Intuitively, this corresponds to the reconstruction matrix of Shamir's secret sharing scheme for $t-1$ parties. By the security of Shamir's scheme, it must be impossible to find a solution to this system of equations. This intuition was formalized, e.g., in [4, Claim 2] or in [18, Appendix E], showing that the above system does not have a solution, and thus f' must be independent of $\mathbf{C}'$. ∎

By Claims E.6, E.7, and E.8, the master theorem applies directly to $(\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_T, f)$ and yields

$$\left|\Pr\left[\mathsf{Game}_{2,\mathcal{A}}(1^\lambda, n, t) = 1\right] - \frac{1}{2}\right| \leq \frac{(q+v+2)^2 d}{2p},$$

where $q \in \mathsf{poly}(\lambda)$ is the number of $\mathcal{A}$'s generic group oracle queries, $v = (2n)(n-1) + 4n + t + 3$ is the number of handles in $\mathcal{A}$'s view, and $d = 2n + 4$ is the degree bound from Claim E.7. Combining with Claims E.4 and E.5 allows us to derive the following bound via the triangle inequality:

$$\mathcal{A}_{\mathcal{A}}^{(t,n)\text{-Int-EBDDH}}(\lambda) = \left|\Pr\left[\mathsf{Game}_{0,\mathcal{A}}(1^\lambda, n, t) = 1\right] - \tfrac{1}{2}\right| \leq \frac{n+2}{p} + \frac{(q+v+2)^2 d}{2p},$$

■