Risky Traitor Tracing and New Differential Privacy Negative Results

Rishab Goyal[*] Venkata Koppula[†] Andrew Russell[‡] Brent Waters[§]

February 27, 2018

Abstract

In this work we seek to construct collusion-resistant traitor tracing systems with small ciphertexts from standard assumptions that also move toward practical efficiency. In our approach we will hold steadfast to the principle of collusion resistance, but relax the requirement on catching a traitor from a successful decoding algorithm. We define a f-risky traitor tracing system as one where the probability of identifying a traitor is $f(\lambda, n)$ times the probability a successful box is produced. We then go on to show how to build such systems from prime order bilinear groups with assumptions close to those used in prior works. Our core system achieves, for any $k > 0$, $f(\lambda, n) \approx \frac{k}{n+k-1}$ where ciphertexts consists of $(k + 4)$ group elements and decryption requires $(k + 3)$ pairing operations.

At first glance the utility of such a system might seem questionable since the f we achieve for short ciphertexts is relatively small. Indeed an attacker in such a system can more likely than not get away with producing a decoding box. However, we believe this approach to be viable for four reasons:

1. A risky traitor tracing system will provide deterrence against risk averse attackers. In some settings the consequences of being caught might bear a high cost and an attacker will have to weigh his utility of producing a decryption D box against the expected cost of being caught.
2. Consider a broadcast system where we want to support low overhead broadcast encrypted communications, but will periodically allow for a more expensive key refresh operation. We refer to an adversary produced algorithm that maintains the ability to decrypt across key refreshes as a persistent decoder. We show how if we employ a risky traitor tracing systems in this setting, even for a small f, we can amplify the chances of catching such a "persistent decoder" to be negligibly close to 1.
3. In certain resource constrained settings risky traitor tracing provides a best tracing effort where there are no other collusion-resistant alternatives. For instance, suppose we had to support 100K users over a radio link that had just 10KB of additional resources for extra ciphertext overhead. None of the existing $\sqrt{N}$ bilinear map systems can fit in these constraints. On the other hand a risky traitor tracing system provides a spectrum of tracing probability versus overhead tradeoffs and can be configured to at least give some deterrence in this setting.
4. Finally, we can capture impossibility results for differential privacy from $\frac{1}{n}$-risky traitor tracing. Since our ciphertexts are short ($O(\lambda)$), we get the negative result which matches what one would get plugging in the obfuscation based tracing system Boneh-Zhandry [BZ14] solution into the prior impossibility result of Dwork et al. [DNR+09].

[*]University of Texas at Austin. Email: rgoyal@cs.utexas.edu.

[†]University of Texas at Austin. Email: kvenkata@cs.utexas.com.

[‡]University of Texas at Austin. Email: ahr@cs.utexas.com.

[§]University of Texas at Austin. Email: bwaters@cs.utexas.edu. Supported by NSF CNS-1228599 and CNS-1414082, DARPA SafeWare, Microsoft Faculty Fellowship, and Packard Foundation Fellowship.

1 Introduction

A traitor tracing [CFN94] system is an encryption system in which a setup algorithm produces a public key pk, master secret key msk and n private keys $\mathsf{sk}_1, \mathsf{sk}_2, \ldots, \mathsf{sk}_n$ that are distributed to n user devices. One can encrypt a message m using the public key to produce a ciphertext ct which can be decrypted using any of the private keys; however, is inaccessible by an attacker that is bereft of any keys. The tracing aspect comes into play if we consider an attacker that corrupts some subset $S \subseteq \{1, \ldots, n\}$ of the devices and produces a decryption algorithm D that decrypts ciphertext with some non-negligible probability $\epsilon(\lambda)$ where λ is the security parameter. An additional Trace algorithm will take as input the master secret key msk and with just oracle access to D will identify at least one user from the corrupted set S (and no one outside it). Importantly, any secure system must be able to handle attackers that will construct D in an arbitrary manner including using techniques such as obfuscation.

While the concept of traitor tracing was originally motivated by the example of catching users that created pirate decoder boxes in broadcast TV systems, there are several applications that go beyond that setting. For example ciphertexts could be encryptions of files stored on cloud storage. Or one might use a broadcast to transmit sensitive information to first responders on an ad-hoc deployed wireless network. In addition, the concepts and techniques of traitor tracing have had broader impacts in cryptography and privacy. Most notably Dwork et al. [DNR+09] showed that the existence of traitor tracing schemes leads to certain impossibility results in the area of differential privacy [DMNS06]. Briefly, they consider the problem of constructing a "sanitizer" $\mathcal{A}$ that takes in a database $x_1, \ldots, x_n$ of entries and wishes to efficiently produce a sanitized summary of database that can evaluate a set of predicate queries on the database. The sanitized database should both support giving an average of answers without too much error and the database should be differentially private in that no one entry should greatly impact the output of the sanitization process. The authors show that an efficient solution to such a problem is impossible to achieve (for certain parameters) assuming the existence of a (collusion resistant) traitor tracing system. The strength of their negative results is directly correlated with the size of ciphertexts in the traitor tracing system.

A primary obstacle in building traitor tracing systems is achieving (full) collusion resistance. There have been several proposals [BF99, NP00, KY02, Sir06, BP08, BN08, KP10] for building systems that are k-collusion resistant where the size of the ciphertexts grows as some polynomial function of k. These systems are secure as long as the number of corrupted keys $|S| \leq k$; however, if the size of the corrupted set exceeds k the attacker will be able to produce a decryption box that is untraceable. Moreover, the collusion bound of k is fixed at system setup so an attacker will know how many keys he needs to exceed to beat the system. In addition, the impossibility results of Dwork et al. [DNR+09] only apply for fully collusion resistant encryption systems. For these reasons we will focus on collusion resistant systems in the rest of the paper.

The existing approaches for achieving collusion resistant broadcast encryption can be fit in the framework of Private Linear Broadcast Encryption (PLBE) introduced by Boneh, Sahai and Waters [BSW06]. In a PLBE system the setup algorithm takes as input a security parameter λ and the number of users n. Like a traitor tracing system it output a public key pk, master secret key msk and n private keys $\mathsf{sk}_1, \mathsf{sk}_2, \ldots, \mathsf{sk}_n$ where a user with index j is given key sk_j. Any of the private keys is capable of decrypting a ciphertext ct created using pk. However, there is an additional $\mathsf{TrEncrypt}$ algorithm that takes in the master secret key, a message and an index i. This produces a ciphertext that only users with index $j \geq i$ can decrypt. Moreover, any adversary produced decryption box D that was created with a set of S where $i \notin S$ would not be able to distinguish between encryption to index i or $i+1$. These properties lead to a tracing system where the tracer measures for each index the probability that D decrypts a ciphertext encrypted (using $\mathsf{TrEncrypt}$) for that index and reports all indices i where there is a significant discrepancy between i and $i+1$. These properties imply that such a PLBE based traitor tracing system will catch at least one user in S with all but negligible probability and not falsely accuse anyone in S.

The primary difficulty in achieving collusion resistant traitor tracing is to do so with short ciphertext size. There are relatively few approaches for achieving this goal. First, one can achieve PLBE in a very simple way from public key encryption. Simply create n independent public and private key pairs from the PKE system and lump all the individual public keys together as the PLBE public key. To encrypt one just encrypts to each sub public key in turn. The downside of this method is that the ciphertext size grows as

$O(n \cdot \lambda)$ as each of the n users need their own slot in the PLBE ciphertext. If one plugs this into the Dwork et al. [DNR+09] impossibility result it rules out systems with a query set $\mathcal{Q}$ of size $2^{O(n \cdot \lambda)}$ or larger. Boneh, Sahai and Waters [BSW06] showed how ciphertexts in a PLBE system can be compressed to $O(\sqrt{n} \cdot \lambda)$ using bilinear maps of composite order. Future variants [GKSW10, Fre10] moved this to the decision linear assumption in prime order groups. While this was an improvement and worked under standard assumptions, there was still a large gap between this and the ideal case where ciphertext size has only polylogarithmic dependence on n.

To achieve really short ciphertexts one needs to leverage heavier tools such as collusion resistant functional encryption or indistingishability obfuscation [BGI+01, GGH+13]. For instance, a simple observation shows that one can make a PLBE scheme directly from a collusion resistant FE scheme such as the [GGH+13]. Boneh and Zhandry [BZ14] gave a construction of PLBE from indistinguishability obfuscation. These two approaches get ciphertexts that grow proportionally to $\log n$ and thus leading to differential privacy impossibility results with smaller query sets of size $n \cdot 2^{O(\lambda)}$. However, general functional encryption and indistinguishability obfuscation candidates currently rely on multilinear map candidates, many of which have been broken and the security of which is not yet well understood. In addition, the actual decryption time resulting from using obfuscation is highly impractical.

Our Results. In this work we seek to construct collusion resistant traitor tracing systems with small ciphertexts from standard assumptions geared towards practical efficiency. In our approach we will hold steadfast to the principle of collusion resistance, but relax the requirement on catching a traitor from a successful decoding algorithm. We define a f-risky traitor tracing system as one where the probability of identifying a traitor is $f(\lambda, n)$ times the probability a successful box is produced. We then go on to show how to build such systems from prime order bilinear groups. Our core system achieves $f(\lambda, n) \approx \frac{k}{n+k-1}$ where ciphertexts consist of $(k+4)$ group elements and decryption requires $(k+3)$ pairing operations, where $k > 0$ is a system parameter fixed at setup time. For the basic setting, i.e. $k = 1$, this gives us a success probability of $\frac{1}{n}$, ciphertext consisting of 5 group elements, and decryption requiring just 4 pairing operations in primer order groups.[1] In addition, we show a generic way to increase f by approximately a factor of c at the cost of increasing the size of the ciphertext and decryption time also by a factor of c.

At first glance the utility of such a system might seem questionable since the function f we achieve for short ciphertexts is relatively small. Indeed an attacker in such a system can more likely than not get away with producing a decoding box. However, we believe this approach to be viable for four reasons:

1. A risky traitor tracing system will provide deterrence against risk averse attackers. In some setting the consequences of being caught might bear a high cost and an attacker will have to weigh his utility of producing a decryption D box against the expected cost of being caught.

2. Consider a broadcast system where we want to support low overhead broadcast encrypted communications, but will periodically allow for a more expensive key refresh operation. We refer to an adversary produced algorithm that maintains the ability to decrypt across key refreshes as a persistent decoder We show how if we employ a risky traitor tracing systems in this setting, even for a small f, we can amplify the chances of catching such a "persistent decoder" to be negligibly close to 1. We discuss this further in our technical overview.

3. In certain resource constrained settings risky traitor tracing provides a best tracing effort where there are no other collusion-resistant alternatives. For instance, suppose we had to support 100K users over a radio link that had just 10KB of additional resources for extra ciphertext overhead. None of the existing $\sqrt{N}$ bilinear map systems [BSW06, BW06, GKSW10, Fre10] can fit in these constraints. On the other hand a risky traitor tracing system provides a spectrum of tracing probability versus overhead tradeoffs and can be configured to at least give some deterrence in this setting.

[1] In addition to our construction from prime-order bilinear groups, we also provide a construction from composite order bilinear groups where ciphertexts consist of three group elements and decryption requires two pairing operations only.

4. Finally, we show that the argument of Dwork et al. applies to $\frac{1}{n}$-risky traitor tracing. Interestingly, when we structure our argument carefully we can achieve the same negative results as when it is applied to a standard traitor tracing system. Since our ciphertexts are short ($O(\lambda)$), we get the negative result which matches what one would get plugging in the obfuscation based tracing system Boneh-Zhandry [BZ14] solution into the prior impossibility result of Dwork et al. [DNR+09].

1.1 Technical Overview

In this section, we give a brief overview of our technical approach. We start by discussing the definitional work. That is, we discuss existing traitor tracing definitions, mention their limitations and propose a stronger (and possibly more useful) definition, and finally introduce a weaker notion of traitor tracing which we call *risky* traitor tracing. Next, we describe our construction for risky traitor tracing from bilinear maps. Lastly, we discuss the differential privacy negative results implied by existence of risky traitor tracing schemes.

Definitional Work. A traitor tracing system consists of four poly-time algorithms — Setup, Enc, Dec, and Trace. The setup algorithm takes as input security parameter λ, and number of users n and generates a public key pk, a master secret key msk, and n private keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_n$. The encrypt algorithm encrypts messages using pk and the decrypt algorithm decrypts a ciphertext using any one of the private keys sk_i. The tracing algorithm takes msk as input and is given a black-box oracle access to a pirate decoder D. It either outputs a special failure symbol $\perp$, or an index $i \in \{1, \ldots, n\}$ signalling that the key sk_i was used to create the pirate decoder.

Traditionally, a traitor tracing scheme is required to satisfy two security properties. First, it must be $\mathsf{IND\text{-}CPA}$ secure, i.e. any PPT adversary, when given no private keys, should not be able to distinguish between encryptions of two different messages. Second, it is required that if an adversary, given private keys $\{\mathsf{sk}_i\}_{i \in S}$ for any set S of its choice, builds a good pirate decoding box D (that is, a decoding box that can can decrypt encryptions of random messages with non-negligible probability), then the trace algorithm should be able to catch one of the private keys used to build the pirate decoding box. Additionally, the trace algorithm should not falsely accuse any user with non-negligible probability. This property is referred to as secure traitor tracing.

Now a limitation of the traitor tracing property as traditionally defined is that a pirate box is labeled as a *good decoder* only if it extracts the entire message from a non-negligible fraction of ciphertexts.[2] In numerous practical scenarios such a definition could be useless and problematic. For instance, consider a pirate box that can always decrypt encryptions of messages which lie in a certain smaller set but does not work on others. If the size of this special set is negligible, then it won't be a good decoder as per existing definitions, but might still be adversarially useful in practice. There are also other reasons why the previous definitions of traitor tracing are problematic (see Section 3.2 for more details). To this end, we use an indistinguishability-based secure-tracing definition, similar to that used in [NWZ16], in which a pirate decoder is labeled to a good decoder if it can distinguish between encryptions of messages chosen by the adversary itself. We discuss this in more detail in Section 3.2.

In this work, we introduce a weaker notion of traitor tracing called f-*risky* traitor tracing, where f is a function that takes the security parameter λ and number of users n as inputs. The syntax as well as $\mathsf{IND\text{-}CPA}$ security requirement is identical to that of standard traitor tracing schemes. The difference is in the way security of tracing traitors is defined. In an f-risky system, we only require that the trace algorithm must catch a traitor with probability at least $f(\lambda, n)$ whenever the adversary outputs a good decoder. This property is referred to as f-risky secure traitor tracing. Note that a 1-risky traitor tracing scheme is simply a standard traitor tracing scheme, and as f decreases, this progressively becomes weaker.

Constructing Risky Traitor Tracing from Bilinear Maps. As mentioned before, our main construction is based on prime order bilinear groups, and leads to a $\frac{k}{n+k-1}$-risky traitor tracing where k is chosen at setup time. However, for ease of technical exposition we start with a simpler construction that uses

[2]The tracing algorithm only needs to work when the pirate box is a good decoder.

composite order bilinear groups and leads to $\frac{1}{n}$-risky traitor tracing scheme. This scheme conveys the basic idea and will serve as a basis for our prime order construction.

Let $\mathbb{G}, \mathbb{G}_T$ be groups of order $N = p_1 p_2 p_3 p_4$ such that there exists a bilinear mapping $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ (that is, a mapping which maps (g^a, g^b) to $e(g, g)^{a \cdot b}$ for all $a, b \in \mathbb{Z}_N$). Since these groups are of composite order, $\mathbb{G}$ has subgroups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, \mathbb{G}_4$ of prime order p_1, p_2, p_3 and p_4 respectively. Moreover, pairing any element in $\mathbb{G}_i$ with an element in $\mathbb{G}_j$ (for $i \neq j$) results in the identity element (we will say that elements in $\mathbb{G}_i$ and $\mathbb{G}_j$ are orthogonal to each other).

At a high level, our construction works as follows. There are three key-generation algorithms: 'less-than' key-generation, 'equal' key-generation and 'greater-than' key-generation. Similarly, we have three encryption algorithms : 'standard' encryption, 'less-than' encryption and 'less-than-equal' encryption. Out of these encryption algorithms, the 'less-than' and 'less-than-equal' encryptions require the master secret key, and are only used for tracing traitors. The decryption functionality can be summarized by Table 1.

	'less-than' keygen	'equal' keygen	'greater-than' keygen
standard enc	✓	✓	✓
'less-than' enc	✗	✓	✓
'less-than-equal' enc	✗	✗	✓

Table 1: Decryption functionality for different encryption/key-generation algorithms. The symbol ✓ denotes that decryption works correctly, while ✗ denotes that decryption fails.

The master secret key consists of a 'cutoff' index i chosen uniformly at random from $\{1, \ldots, n\}$. For any index $j < i$, it uses the 'less-than' key-generation algorithm to generate keys. For $j > i$, it uses the 'greater-than' key-generation algorithm, and for $j = i$, it uses the 'equal' key-generation algorithm. The ciphertext for a message m is a 'standard' encryption of m. From Table 1, it is clear that decryption works. The trace algorithm tries to identify if the cutoff index i is used by the pirate box D. It first checks if D can decrypt 'less-than' encryptions. If so, then it checks if D can decrypt 'less-than-equal' encryptions. If D works in the 'less-than' case, but not in the 'less-than-equal' case, then the trace algorithm identifies index i as one of the traitors.

Let us now look at how the encryption/key generation algorithms work at a high level. The public key in our scheme consists of $g_1 \in \mathbb{G}_1$ and $e(g_1, g_1)^\alpha$, while the master secret key has the cut-off index i, element α, as well as generators for all subgroups of $\mathbb{G}$. The 'less-than' keys are set to be $g_1^\alpha \cdot w_3 \cdot w_4$, where w_3, w_4 are random group elements from $\mathbb{G}_3, \mathbb{G}_4$ respectively. The 'equal' key is $g_1^\alpha \cdot w_2 \cdot w_4$, where $w_2 \leftarrow \mathbb{G}_2, w_4 \leftarrow \mathbb{G}_4$. Finally, the 'greater-than' key has no $\mathbb{G}_2$ or $\mathbb{G}_3$ terms, and is set to be $g_1^\alpha \cdot w_4$.

The 'standard' encryption of message m is simply $(m \cdot e(g_1, g_1)^{\alpha \cdot s}, g_1^s)$. In the 'less-than' and 'less-than-equal' ciphertexts, the first component is computed similarly but the second component is modified. For 'less-than' encryptions, the ciphertext is $(m \cdot e(g_1, g_1)^{\alpha \cdot s}, g_1^s \cdot h_3)$, where h_3 is a uniformly random group element in $\mathbb{G}_3$. For 'less-than-equal' encryptions the ciphertext is $(m \cdot e(g_1, g_1)^{\alpha \cdot s}, g_1^s \cdot h_2 \cdot h_3)$, where h_2 and h_3 are uniformly random group elements in $\mathbb{G}_2$ and $\mathbb{G}_3$ respectively.

To decrypt a ciphertext $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2)$ using a key K, one must compute $\mathsf{ct}_1 / e(\mathsf{ct}_2, K)$. It is easy to verify that the keys and encryptions follow the decryption behavior described in Table 1. For instance, an 'equal' key $K = g_1^\alpha \cdot w_2 \cdot w_4$ can decrypt a 'less-than' encryption $(m \cdot e(g_1, g_1)^{\alpha \cdot s}, g_1^s \cdot h_3)$ because $e(\mathsf{ct}_2, K) = e(g_1, g_1)^{\alpha \cdot s}$. However, an 'equal' key cannot decrypt a 'less-than-equal' ciphertext $\mathsf{ct} = (m \cdot e(g_1, g_1)^{\alpha \cdot s}, g_1^s \cdot h_2)$ because $e(\mathsf{ct}_2, K) = e(g_1, g_1)^{\alpha \cdot s} \cdot e(h_2, w_2)$.

Given this construction, we need to prove two claims. First, we need to show that no honest party is implicated by our trace algorithm; that is, if an adversary does not receive key for index i, then the trace algorithm must not output index i. We show that if an adversary does not have key for index i, then the pirate decoding box must not be able to distinguish between 'less-than' and 'less-than-equal' encryptions (otherwise we can break the subgroup-decision assumption on composite order bilinear groups). Next, we show that if an adversary outputs a pirate decoding box that works with probability ρ, then we can identify a traitor with probability ρ/n. To prove this, we show that if ρ_i denotes the probability that the adversary outputs a ρ-functional box and i is the cutoff-index, then the sum of all these ρ_i quantities is close to ρ. The above scheme is formally described later in Appendix C along with a detailed security proof. Next we move on to our risky traitor tracing construction from prime order bilinear groups.

Moving to Prime Order Bilinear Maps and $\frac{k}{n+k-1}$-Risky. The starting point for building $\frac{k}{n+k-1}$-risky traitor tracing scheme from prime order bilinear groups is the aforementioned scheme. Now to increase the success probability of the tracing algorithm by a factor k, we increase the types of secret keys and ciphertexts from 3 to $k+2$ such that the decryptability of ciphertexts w.r.t. secret keys can again be described as an upper-triangular matrix of dimension $k+2$ as follows.

	'$< w$' keygen	'$= w$' keygen	'$= w+1$' keygen	$\cdots$	'$= w+k-1$' keygen	'$\geq w+k$' keygen
standard enc	✓	✓	✓	$\cdots$	✓	✓
'$< w$' enc	✗	✓	✓	$\cdots$	✓	✓
'$< w+1$' enc	✗	✗	✓	$\cdots$	✓	✓
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\ddots$	$\vdots$	$\vdots$
'$< w+k-1$' enc	✗	✗	✗	$\cdots$	✓	✓
'$< w+k$' enc	✗	✗	✗	$\cdots$	✗	✓

Table 2: New Decryption Functionality.

The basic idea will similar to the one used previously, except now we choose a cutoff window $W = \{w, w+1, \ldots, w+k-1\}$ of size k uniformly at random. (Earlier the window had size 1, that is we choose a single index.) The first $w-1$ users are given '$< w$' keys. For $w \leq j < w+k$, the j^{th} user gets '$= j$' key, and rest of the users get the '$\geq w+k$' keys. The remaining idea is similar to what we used which is that the tracer estimates the successful decryption probability for a decoder D on all the special index encryptions (i.e., '$< j$' encryptions), and outputs the indices of all those users where there is a gap in decoding probability while moving from type '$< j$' to '$< j+1$'.

Now instead of directly building a scheme that induces such a decryption functionality, we provide a general framework for building risky traitor tracing schemes. In this work, we introduce a new intermediate primitive called Mixed Bit Matching Encryption (mBME) and show that it is sufficient to build risky traitor tracing schemes. In a mBME system, the secret keys and ciphertexts are associated with bit vectors $\mathbf{x}, \mathbf{y} \in \{0,1\}^\ell$ (respectively) for some ℓ. And decryption works whenever $f(\mathbf{x}, \mathbf{y}) = 1$ where f computes an 'AND-of-ORs' over vectors $\mathbf{x}, \mathbf{y}$ (i.e., for every $i \leq \ell$, either $\mathbf{x}_i = 1$ or $\mathbf{y}_i = 1$). Using the public parameters, one could encrypt to the 'all-ones' vector, and using the master secret key one could sample a ciphertext (or secret key) for any vector. For security, we require that the ciphertexts and the secret keys should not reveal non-trivial information about their associated vectors. In other words, the only information an adversary learns about these vectors is by running the decryption algorithm. In the sequel, we provide a generic construction of risky traitor tracing from a mBME scheme, and also give a construction of mBME scheme using prime order bilinear groups. They are described in detail later in Sections 5 and 6.

Finally, we also provide a performance evaluation of our risky traitor tracing scheme in Section 7.

Relation to BSW traitor tracing scheme. Boneh, Sahai and Waters [BSW06] constructed a (fully) collusion-resistant traitor tracing scheme with $O(\sqrt{n} \cdot \lambda)$ size ciphertexts. The BSW construction introduced the *private linear broadcast encryption* (PLBE) abstraction, showed how to build traitor tracing using PLBE, and finally gave a PLBE construction using composite-order bilinear groups.

Our framework deviates from the PLBE abstraction in that we support encryptions to only $k+1$ adjacent indices (that is, if w is starting index of the cutoff window, then we support encryptions to either $w, \ldots, w+k$) and index 0. As a result, the trace algorithm can only trace an index in the window $w, \ldots, w+k$. The main difficulty in our proof argument is that encrypting to index j is not defined for indices oustide the cutoff window, i.e. $j \notin \{0, w, w+1, \ldots, w+k\}$. As a result, we need to come up with a new way to link success probabilities across different setups and weave these into an argument.

Negative Results for Differential Privacy. Given a database $D = (x_1, x_2, \ldots, x_n) \in \mathcal{X}^n$, in which each row represents a single record of some sensitive information contributed by an individual and each record is an element in the data universe $\mathcal{X}$, the problem of privacy-preserving data analysis is to allow statistical analyses of D while protecting the privacy of individual contributors. The problem is formally defined in the literature by representing the database with a *sanitized* data structure s that can be used to answer all queries q in some query class $\mathcal{Q}$ with reasonable accuracy, with the restriction that the sanitization of any two databases D, D' which differ at only a single position are indistinguishable. In this work, we will focus on counting (or statistical) queries. Informally, a counting query q on a database D tells what fraction of records in D satisfy the property associated with q.

Dwork et al. [DNR+09] first showed that secure traitor tracing schemes can be used to show hardness results for efficient differentially private sanitization. In their hardness result, the data universe is the private key space of traitor tracing scheme and the query space is the ciphertext space. A database consists of n private keys and each query is associated with either an encryption of 0 or 1. Formally, for a ciphertext ct, the corresponding query q_{ct} on input a private key sk outputs the decryption of ct using sk. They show that if the underlying traitor tracing scheme is secure, then there can not exist sanitizers that are simultaneously accurate, differentially private, and efficient. At a very high level, the idea is as follows. Suppose there exists an efficient sanitizer A that, on input $D = (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)$ outputs a sanitization s. The main idea is to use sanitizer A to build a pirate decoding box such that the tracing algorithm falsely accuses a user with non-negligible probability, thereby breaking secure traitor traitor property. Concretely, let $\mathcal{B}$ be an attacker on the secure tracing property that works as follows — $\mathcal{B}$ queries for private keys of all but i^{th} party, and then uses sanitizer A to generate sanitization s of the database containing all the queried private keys, and finally it outputs the pirate decoding box as the sanitization evaluation algorithm which has s hardwired inside and on input a ciphertext ouputs its evaluation given sanitization s.[3]

To prove that the tracing algorithm outputs i (with non-negligible probability) given such a decoding box, Dwork et al. crucially rely on the fact that A is differentially private. First, they show that if an adversary uses all private keys to construct the decoding box, then the tracing algorithm always outputs an index and never aborts.[4] Then, they argue that there must exist an index i such that tracing algorithm outputs i with probability $p \geq 1/n$. Finally, to complete the claim they show that even if i^{th} key is removed from the database, the tracing algorithm will output i with non-negligible probability since the sanitizer is differentially private with parameters $\epsilon = O(1)$ and $\delta = o(1/n)$.

In this work, we show that their analysis can be adapted to risky traitor tracing as well. Concretely, we show that f-risky secure traitor tracing schemes can be used to show hardness results for efficient differentially private sanitization, where f directly relates to the differential privacy parameters. At a high level, the proof strategy is similar, i.e. we also show that an efficient sanitizer could be used to build a good pirate decoding box. The main difference is that now we can only claim that if an adversary uses all private keys to construct the decoding box, then (given oracle access to the box) the tracing algorithm outputs an index with probability at least f, i.e. the trace algorithm could potentially abort with non-negligible probability. Next, we can argue that there must exist an index i such that tracing algorithm outputs i with probability $p \geq f/n$. Finally, using differential privacy of A we can complete the argument. An important caveat in the proof is that since the lower bounds in the probability terms have an additional multiplicative factor of f, thus f-risky traitor tracing could only be used to argue hardness of differential privacy with slightly lower values of parameter δ, i.e. $\delta = o(f/n)$.

However, we observe that if the risky traitor tracing scheme additionally satisfies what we call "singular trace" property, then we could avoid the $1/n$ loss. Informally, a risky scheme is said to satisfy the singular trace property if the trace algorithm always outputs either a fixed index or the empty set. One could visualize the fixed index to be tied to the master secret and public keys. Concretely, we show that f-risky traitor tracing with singular trace property implies hardness of differential privacy for $\delta = o(f)$, thereby matching that achieved by previous obfuscation based result of [BZ14]. We describe our hardness result in detail in

[3]Technically, the decoding box must round the output of evaluation algorithm in order to remove evaluation error.

[4]In the full proof, one could only argue that tracing algorithm outputs an index with probability at least $1 - \beta$ where β is the accuracy parameter of sanitizer A.

Section 8.2.

Amplifying the Probability of Tracing — Catching Persistent Decoders. While an f-risky traitor tracing system by itself gives a small probability of catching a traitor, there can be ways to deploy it that increase this dramatically. We discuss one such way informally here.

Consider a broadcast system where we want to support low overhead broadcast encrypted communications, but will periodically allow for a more expensive key refresh operation. Suppose that we generate the secret keys $\mathsf{sk}_1, \mathsf{sk}_2, \ldots, \mathsf{sk}_n$ for a risky traitor tracing system and in addition generate standard secret keys $\mathrm{SK}_1, \ldots, \mathrm{SK}_n$. In this system an encryptor can use the traitor tracing public key pk to compute a ciphertext. A user i will use secret key sk_i to decrypt. The system will allow this to continue for a certain window of time. (Note during the window different ciphertexts may be created by different users.) Then at some point in time the window will close and a new risky tracing key pk' and secret keys $\mathsf{sk}'_1, \mathsf{sk}'_2, \ldots, \mathsf{sk}'_n$ will be generated. The tracing secret keys will be distributed by encrypting each sk'_i under the respective permanent secret key SK_i. And the encryptors will be instructed to only encrypt using the new public key pk'. This can continue for an arbitrary number of windows followed by key refreshes. Note that each key refresh requires $O(n\lambda)$ size communication.

Consider an attacker that wishes to disperse a stateless decoder D that is capable of continuing to work through multiple refresh cycles. Such a "persistent decoder" can be traced with very high probability negligibly close to 1. The tracing algorithm must simply give it multiple key refreshes followed by calls to the Trace algorithm and by the risky property it will eventually pick one that can trace one of the contributors.

We emphasize that care must be taken when choosing the refresh size window. If the window is too small the cost of key refreshes will dominate communication — in one extreme if a refresh happens at the frequency that ciphertexts are created then the communication is as bad as the trivial PLBE system. In addition, dispersing new public keys very frequently can be an issue. On the other hand if a refresh window is very long, then an attacker might decide there is value in producing a decoding box that works only for the given window and we are back to having only an $f(\lambda, n)$ chance of catching him.

1.2 Additional Related Work

Our traitor tracing system allows for public key encryption, but requires a master secret key to trace users as do most works. However, there exists exceptions [Pfi96, PW97, WHI01, KY03, CPP05, BW06, BZ14] where the tracing can be done using a public key. In a different line of exploration, Kiayias and Yung [KY02] argue that a traitor tracing system with higher overhead can be made "constant rate" with long enough messages. Another interesting point in the space of collusion resistant systems is that of Boneh and Naor [BN08]. They show how to achieve short ciphertext size, but require private keys that grow quadratically in the number of users as $O(n^2\lambda)$. In addition, this is only achievable assuming a perfect decoder. If the decoder D works with probability δ then the secret key grows to $O(n^2\lambda/\delta^2)$. Furthermore, the system must be configured a-priori with a specific δ value and once it is set one will not necessarily be able to identify a traitor from a box D that works with smaller probability. Such systems have been called threshold traitor tracing systems [NP98, CFNP00]. Both [NP98, CFNP00] provide combinatorial and probabilistic constructions in which the tracing algorithm is guaranteed to work with high probability, and to trace t traitors they get private keys of size $O(t \cdot \log n)$. In contrast we can capture any traitor strategy that produces boxes that work with any non-negligible function $\epsilon(\lambda)$. Chor et al. [CFNP00] also considered a setting for traitor tracing in which the tracing algorithm only needs to correctly trace with probability $1 - p$, where p could the scheme parameter. However, this notion has not been formally defined or explored since then.

Dwork et al. [DNR+09] first showed that existence of collusion resistant traitor tracing schemes implies hardness results for efficient differentially private sanitization. In their hardness result, the database consists of n secret keys and each query is associated with an encryption of 0/1. Thus, the size of query space depends on the size of ciphertexts. Instantiating the result of Dwork et al. with the traitor tracing scheme of Boneh et al. [BSW06], we get that under assumptions on bilinear groups, there exist a distribution on databases of size n and a query space of size $O(2^{\sqrt{n} \cdot \lambda})$ such that it is not possible to efficiently sanitize the database in a differentially private manner.

Now the result of Dwork et al. gives hardness of *one-shot* sanitization. A one-shot sanitizer is supposed to produce a summary of an entire database from which approximate answers to any query in the query set could be computed. A weaker setting could be where we consider interactive sanitization, in which the queries are fixed and given to the sanitizer as an additional input and the sanitizer only needs to output approximate answers to all those queries instead of a complete summary. Ullman [Ull13] showed that, under the assumption that one-way functions exist, there is no algorithm that takes as input a database of n records along with an arbitrary set of about $O(n^2)$ queries, and approximately answers each query in polynomial time while preserving differential privacy. Ullman's result differs from the result of Dwork et al. in that it applies to algorithms answering any arbitrary set of $O(n^2)$ queries, whereas Dwork et al. show that it is impossible to sanitize a database with respect to a fixed set of $O(2^{\sqrt{n}\cdot\lambda})$ queries.

Recently a few works [BZ14, KMUZ16] have improved the size of query space for which (one-shot) sanitization is impossible from $O(2^{\sqrt{n}\cdot\lambda})$ to $n \cdot O(2^\lambda)$ to $\mathsf{poly}(n)$.[5] [BZ14] showed the impossibility by first constructing a fully collusion resistant scheme with short ciphertexts, and later simply applying the Dwork et al. result. On the other hand, [KMUZ16] first construct a weakly secure traitor tracing scheme by building on top of PLBE abstraction, and later adapt the Dwork et al. impossibility result for this weaker variant. These works however assume existence of a stronger cryptographic primitive called *indistinguishability-obfuscator* ($i\mathcal{O}$) [BGI+01, GGH+13]. Currently we do not know of any construction of $i\mathcal{O}$ from a standard cryptographic assumption. In this work, we are interested in improving the state-of-the-art hardness results in differential privacy based on standard assumptions.

More recent related work. In an independent and concurrent work, Kowalczyk, Malkin, Ullman and Wichs [KMUW17] gave similar differential privacy negative results from any functional encryption system for comparisons that supported two ciphertext and an unbounded number of secret keys. And showed how to realize this from one way functions. The negative results they achieve are similar to ours, but apply for slightly smaller database sizes. The focus of their work is on negative results for differential privacy, whereas our risky tracing framework has both positives application as well as differential privacy impossibility results.

Subsequent to our work, Goyal, Koppula and Waters [GKW18] gave a collusion resistant tracing system from the Learning with Errors assumption where the ciphertext size grows polynomially in $\lambda, \lg(N)$.

2 Preliminaries

Notations. For any set $\mathcal{X}$, let $x \leftarrow \mathcal{X}$ denote a uniformly random element drawn from the set $\mathcal{X}$. Given a PPT algorithm D, let A^D denote an algorithm $\mathcal{A}$ that uses D as an oracle (that is, A sends queries to D, and for each query x, it receives $D(x)$). Throughout this paper, we use PPT to denote probabilistic polynomial-time. We will use lowercase bold letters for vectors (e.g. $\mathbf{v}$), and we will sometimes represent bit vectors $\mathbf{v} \in \{0,1\}^\ell$ as bit-strings of appropriate length.

2.1 Assumptions

In this work, we will be using bilinear groups. Let $\mathsf{Grp\text{-}Gen}$ be a PPT algorithm that takes as input security parameter λ (in unary), and outputs a λ-bit prime p, an efficient description of groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of order p, generators $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ and an efficient non-degenerate bilinear mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ (that is, $e(g_1, g_2) \neq 1_{G_T}$, and for all $a, b \in \mathbb{Z}_p$, $e(g_1^a, g_2^b) = e(g_1, g_2)^{a\cdot b}$).

We will be using the following assumptions in this work.

Assumption 1. For every PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ s.t. for all $\lambda \in \mathbb{N}$,

$$\Pr\left[b \leftarrow \mathcal{A}\left(\begin{array}{c} \mathsf{params}, \\ g_1^x, g_1^y, g_1^{y\cdot z}, g_2^y, g_2^z, T_b \end{array} \right) : \begin{array}{c} \mathsf{params} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e(\cdot,\cdot)) \leftarrow \mathsf{Grp\text{-}Gen}(1^\lambda); \\ x, y, z, r \leftarrow \mathbb{Z}_p, T_0 = g_1^{x\cdot y\cdot z}, T_1 = g_1^{x\cdot y\cdot z + r}, b \leftarrow \{0,1\} \end{array} \right] \leq 1/2 + \text{negl}(\lambda).$$

[5] In this work, we only focus on the size of query space.

Assumption 2. For every PPT adversary $\mathcal{A}$, there exists a negligible function $\mathrm{negl}(\cdot)$ s.t. for all $\lambda \in \mathbb{N}$,

$$\Pr\left[b \leftarrow \mathcal{A}\left(\begin{array}{c} \mathsf{params}, \\ g_1^y, g_1^z, g_2^x, g_2^y, T_b \end{array} \right) : \begin{array}{c} \mathsf{params} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e(\cdot,\cdot)) \leftarrow \mathsf{Grp\text{-}Gen}(1^\lambda); \\ x, y, z, r \leftarrow \mathbb{Z}_p, T_0 = g_2^{x \cdot y \cdot z}, T_1 = g_2^{x \cdot y \cdot z + r}, b \leftarrow \{0,1\} \end{array} \right] \leq 1/2 + \mathrm{negl}(\lambda).$$

3 Risky Traitor Tracing

In this section, we will first introduce the traditional definition of traitor tracing based on that given by Boneh, Sahai and Waters [BSW06]. We provide a "public key" version of the definition in which the encryption algorithm is public, but the tracing procedure will require a master secret key. Our definition will by default capture full collusion resistance.

A limitation of this definition is that the tracing algorithm is only guaranteed to work on decoders that entirely decrypt encryptions of randomly selected messages with non-negligible probability. We we will discuss why this definition can be problematic and then provide an *indistinguishability* based definition for secure tracing.

Finally, we will present our new notion of *risky* traitor tracing which captures the concept of a trace algorithm that will identify a traitor from a working pirate box with probability close to $f(\lambda, n)$. Our main definition for risky traitor tracing will be a public key one using the indistinguishability; however we will also consider some weaker variants that will be sufficient for obtaining our negative results in differential privacy.

3.1 Public Key Traitor Tracing

A traitor tracing scheme with message space $\mathcal{M}$ consists of four PPT algorithms $\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}$ and Trace with the following syntax:

$(\mathsf{msk}, \mathsf{pk}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$: The setup algorithm takes as input the security parameter λ, number of users n, and outputs a master secret key msk, a public key pk and n secret keys $\mathsf{sk}_1, \mathsf{sk}_2, \ldots, \mathsf{sk}_n$.

$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m \in \mathcal{M})$: The encryption algorithm takes as input a public key pk, message $m \in \mathcal{M}$ and outputs a ciphertext ct.

$y \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: The decryption algorithm takes as input a secret key sk, ciphertext ct and outputs $y \in \mathcal{M} \cup \{\perp\}$.

$S \leftarrow \mathsf{Trace}^D(\mathsf{msk}, 1^y)$: The tracing algorithm takes a parameter $y \in \mathbb{N}$ (in unary) as input, has black box access to an algorithm D, and outputs a set $S \subseteq \{1, 2, \ldots, n\}$.

Correctness For correctness, we require that if ct is an encryption of message m, then decryption of ct using one of the valid secret keys must output m. More formally, we require that for all $\lambda \in \mathbb{N}$, $n \in \mathbb{N}$, $(\mathsf{msk}, \mathsf{pk}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, $m \in \mathcal{M}$, $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m)$ and $i \in \{1, 2, \ldots, n\}$, $\mathsf{Dec}(\mathsf{sk}_i, \mathsf{ct}) = m$.

Security A secure traitor tracing scheme must satisfy two security properties. First, the scheme must be $\mathsf{IND\text{-}CPA}$ secure (that is, any PPT adversary, when given no secret keys, cannot distinguish between encryptions of m_0, m_1). Next, we require that if an adversary, using some secret keys, can build a pirate decoding box, then the trace algorithm should be able to catch at least one of the secret keys used to build the pirate decoding box. In this standard definition, the trace algorithm identifies a traitor if the pirate decoding box works with non-negligible probability in extracting the entire message from an encryption of a random message.

Definition 3.1 ($\mathsf{IND\text{-}CPA}$ security)**.** A traitor tracing scheme $\mathcal{T} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$ is $\mathsf{IND\text{-}CPA}$ secure if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, polynomial $n(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $|\Pr[1 \leftarrow \mathsf{Expt\text{-}IND\text{-}CPA}_{\mathcal{A}}^{\mathcal{T}}(1^\lambda, 1^n)] - 1/2| \leq \mathrm{negl}(\lambda)$, where $\mathsf{Expt\text{-}IND\text{-}CPA}_{\mathcal{T},\mathcal{A}}$ is defined below.

- $(\mathsf{msk}, \mathsf{pk}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)) \leftarrow \mathsf{Setup}(1^\lambda, 1^{n(\lambda)})$
- $(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(\mathsf{pk})$
- $b \leftarrow \{0,1\}$, $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$
- $b' \leftarrow \mathcal{A}_2(\sigma, \mathsf{ct})$. Experiment outputs 1 iff $b = b'$.

Definition 3.2 (Secure traitor tracing)**.** Let $\mathcal{T} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$ a traitor tracing scheme. For any polynomial $n(\cdot)$, non-negligible function $\epsilon(\cdot)$ and PPT adversary $\mathcal{A}$, consider the following experiment $\mathsf{Expt}^{\mathcal{T}}_{\mathcal{A},n,\epsilon}(\lambda)$:

- $\big(\mathsf{msk}, \mathsf{pk}, \big(\mathsf{sk}_1, \ldots, \mathsf{sk}_{n(\lambda)}\big)\big) \leftarrow \mathsf{Setup}(1^\lambda, 1^{n(\lambda)})$.
- $D \leftarrow A^{O(\cdot)}(\mathsf{pk})$
- $S_D \leftarrow \mathsf{Trace}^D(\mathsf{msk}, 1^{1/\epsilon(\lambda)})$.

Here, $O(\cdot)$ is an oracle that has $\{\mathsf{sk}_1, \mathsf{sk}_2, \ldots, \mathsf{sk}_{n(\lambda)}\}$ hardwired, takes as input an index $i \in \{1, 2, \ldots, n(\lambda)\}$ and outputs sk_i. Let S be the set of indices queried by $\mathcal{A}$. Based on this experiment, we will now define the following (probabilistic) events and the corresponding probabilities (which is a function of λ, parameterized by $\mathcal{A}, n, \epsilon$):

- $\mathsf{Good\text{-}Decoder}$: $\Pr[D(\mathsf{ct}) = m \ : \ m \leftarrow \mathcal{M}, \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m)] \geq \epsilon(\lambda)$
  $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Good\text{-}Decoder}]$.
- $\mathsf{Cor\text{-}Tr}$: $S_D \subseteq S \ \wedge \ S_D \neq \emptyset$
  $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Cor\text{-}Tr}]$.
- $\mathsf{Fal\text{-}Tr}$: $S_D \setminus S \neq \emptyset$
  $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Fal\text{-}Tr}]$.

A traitor tracing scheme $\mathcal{T}$ is said to be secure if for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot)$, $p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists negligible functions $\mathrm{negl}_1(\cdot)$, $\mathrm{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$ such that $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \leq \mathrm{negl}_1(\lambda)$ and $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) - \mathrm{negl}_2(\lambda)$.

3.2 Indistinguishability Security Definition for Traitor Tracing Schemes

A limitation of the previous definition is that the tracing algorithm is only guaranteed to work on decoders that *entirely* decrypt a *randomly* selected message with non-negligible probability. This definition can be problematic for the following reasons.

- First, there could be pirate boxes which do not extract the entire message from a ciphertext, but can extract some information about the message underlying a ciphertext. For example, a box could paraphrase English sentences or further compress an image. Such boxes could be very useful to own in practice yet the tracing definition would give no guarantees on the ability to trace them.
- Second, a pirate decoder may not be very successful in decrypting random ciphertexts, but can decrypt encryptions of messages from a smaller set. In practice the set of useful or typical messages might indeed fall in a smaller set.
- Finally, if the message space is small (that is, of polynomial size), then one can always construct a pirate decoder which succeeds with non-negligible probability and can not get caught (the pirate decoder box simply outputs a random message for each decryption query. If $\mathcal{M}$ is the message space, then decryption will be successful with probability $1/|\mathcal{M}|$). Since such a strategy does not use any private keys, it cannot be traced. Therefore the above definition is only sensible for superpolynomial sized message spaces.

To address these issues, we provide a stronger definition, similar to that used in [NWZ16], in which a pirate decoder is successful if it can distinguish between encryptions of messages chosen by the decoder itself. For this notion, we also need to modify the syntax of the Trace algorithm. Our security notion is similar to the one above except that an attacker will output a box D along with two messages (m_0, m_1). If the box D is able to distinguish between encryptions of these two messages with non-negligible probability then the tracing algorithm can identify a corroborating user.

$\mathsf{Trace}^D(\mathsf{msk}, 1^y, m_0, m_1)$: The trace algorithm has oracle access to a program D, it takes as input a master secret key msk, y (in unary) and two messages m_0, m_1. It outputs a set $S \subseteq \{1, 2, \ldots, n\}$.

Definition 3.3 (Ind-secure traitor tracing)**.** Let $\mathcal{T} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$ be a traitor tracing scheme. For any polynomial $n(\cdot)$, non-negligible function $\epsilon(\cdot)$ and PPT adversary $\mathcal{A}$, consider the experiment $\mathsf{Expt\text{-}TT}^{\mathcal{T}}_{\mathcal{A},n,\epsilon}(\lambda)$ defined in Figure 1. Based on this experiment, we will now define the following (probabilistic) events and the corresponding probabilities (which is a function of λ, parameterized by $\mathcal{A}, n, \epsilon$):

- $\mathsf{Good\text{-}Decoder}$: $\Pr[D(\mathsf{ct}) = b \ : \ b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)] \geq 1/2 + \epsilon(\lambda)$
  $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Good\text{-}Decoder}]$.
- $\mathsf{Cor\text{-}Tr}$: $S_D \subseteq S \ \wedge \ S_D \neq \emptyset$
  $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Cor\text{-}Tr}]$.
- $\mathsf{Fal\text{-}Tr}$: $S_D \setminus S \neq \emptyset$
  $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Fal\text{-}Tr}]$.

A traitor tracing scheme $\mathcal{T}$ is said to be ind-secure if for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot)$, $p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists negligible functions $\mathrm{negl}_1(\cdot)$, $\mathrm{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \leq \mathrm{negl}_1(\lambda)$ and $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) - \mathrm{negl}_2(\lambda)$.

Experiment $\mathsf{Expt\text{-}TT}^{\mathcal{T}}_{\mathcal{A},n,\epsilon}(\lambda)$

- $(\mathsf{msk}, \mathsf{pk}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_{n(\lambda)})) \leftarrow \mathsf{Setup}(1^\lambda, 1^{n(\lambda)})$.
- $(D, m_0, m_1) \leftarrow \mathcal{A}^{O(\cdot)}(\mathsf{pk})$
- $S_D \leftarrow \mathsf{Trace}^D(\mathsf{msk}, 1^{1/\epsilon(\lambda)}, m_0, m_1)$.

Here, $O(\cdot)$ is an oracle that has $\{\mathsf{sk}_1, \mathsf{sk}_2, \ldots, \mathsf{sk}_{n(\lambda)}\}$ hardwired, takes as input an index $i \in \{1, 2, \ldots, n(\lambda)\}$ and outputs sk_i. Let S be the set of indices queried by $\mathcal{A}$.

Figure 1: Experiment $\mathsf{Expt\text{-}TT}$

3.3 Risky Traitor Tracing

In this section, we will introduce the notion of risky traitor tracing. The syntax is same as that of ind-secure traitor tracing. However, for security, if the adversary outputs a good decoder, then the trace algorithm will catch a traitor with probability f where f is a function of λ and the number of users.

Definition 3.4 (f-risky secure traitor tracing)**.** Let $f : \mathbb{N} \times \mathbb{N} \rightarrow [0, 1]$ be a function and $\mathcal{T} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$ a traitor tracing scheme. For any polynomial $n(\cdot)$, non-negligible function $\epsilon(\cdot)$ and PPT adversary $\mathcal{A}$, consider the experiment $\mathsf{Expt\text{-}TT}^{\mathcal{T}}_{\mathcal{A},n,\epsilon}(\lambda)$ (defined in Figure 1). Based on this experiment, we will now define the following (probabilistic) events and the corresponding probabilities (which are functions of λ, parameterized by $\mathcal{A}, n, \epsilon$):

- $\mathsf{Good\text{-}Decoder}$: $\Pr[D(\mathsf{ct}) = b \ : \ b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)] \geq 1/2 + \epsilon(\lambda)$
  $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Good\text{-}Decoder}]$.

- $\mathsf{Cor\text{-}Tr}$: $S_D \subseteq S \ \wedge \ S_D \neq \emptyset$
  $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Cor\text{-}Tr}]$.
- $\mathsf{Fal\text{-}Tr}$: $S_D \setminus S \neq \emptyset$
  $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Fal\text{-}Tr}]$.

A traitor tracing scheme $\mathcal{T}$ is said to be f-risky secure if for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot)$, $p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists negligible functions $\text{negl}_1(\cdot)$, $\text{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \leq \text{negl}_1(\lambda)$ and $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) \cdot f(\lambda, n(\lambda)) - \text{negl}_2(\lambda)$.

We also define another interesting property for traitor tracing schemes which we call "singular" trace. Informally, a scheme satisfies it if the trace algorithm always outputs either a fixed index or the reject symbol. The fixed index could depend on the master secret and public keys. Below we define it formally.

Definition 3.5 (Singular Trace)**.** A traitor tracing scheme $\mathcal{T} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$ is said to satisfy *singular trace* property if for every polynomial $n(\cdot)$, $\lambda \in \mathbb{N}$, keys $(\mathsf{msk}, \mathsf{pk}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, there exists an index $i^* \in \{1, \ldots, n\}$ such that for every poly-time algorithm D, parameter $y \in \mathbb{N}$, any two messages m_0, m_1,

$$\Pr[\mathsf{Trace}^D(\mathsf{msk}, 1^y, m_0, m_1) \in \{\{i^*\}, \emptyset\}] = 1,$$

where the probability is taken over random coins of Trace.

3.4 Private Key Traitor Tracing

We will now present different security notions for private key encryption schemes with risky traitor tracing. Here, the master secret key is used for encrypting messages, generating secret keys for parties and tracing traitors. The first security notion will be a private key analog of Definition 3.4, where the adversary also gets encryption queries before it sends the pirate decoding box. In the second notion, the adversary does not get any encryption queries. While this definition is weaker than the first notion (and may not capture practical scenarios), it suffices for our differential privacy application.

First, we will present the syntax for private key traitor tracing. A private key traitor tracing scheme for message space $\mathcal{M}$ consists of the following algorithms.

$(\mathsf{msk}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$: The setup algorithm takes as input the security parameter λ, number of users n, and outputs a master secret key msk and n secret keys $\mathsf{sk}_1, \mathsf{sk}_2, \ldots, \mathsf{sk}_n$.

$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{msk}, m \in \mathcal{M})$: The encryption algorithm takes as input a master secret key pk, message $m \in \mathcal{M}$ and outputs a ciphertext ct.

$y \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: The decryption algorithm takes as input a secret key sk, ciphertext ct and outputs $y \in \mathcal{M} \cup \{\bot\}$.

$S \leftarrow \mathsf{Trace}^D(\mathsf{msk}, 1^y)$: The tracing algorithm takes a parameter $y \in \mathbb{N}$ (in unary) as input, has black box access to an algorithm D, and outputs a set $S \subseteq \{1, 2, \ldots, n\}$.

The correctness property is similar to that in the public key setting.

3.4.1 Security

Definition 3.6 (Private Key f-risky secure traitor tracing)**.** Let $\mathcal{T} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$ be a private key traitor tracing scheme. For any polynomial $n(\cdot)$, non-negligible function $\epsilon(\cdot)$ and PPT adversary $\mathcal{A}$, consider the experiment $\mathsf{Expt\text{-}TT\text{-}priv}^{\mathcal{T}}_{\mathcal{A},n,\epsilon}(\lambda)$ (described in Figure 2). Based on this experiment, we will now define the following (probabilistic) events and the corresponding probabilities (which is a function of λ, parameterized by $\mathcal{A}, n, \epsilon$):

- $\mathsf{Good\text{-}Decoder}$: $\Pr[D(\mathsf{ct}) = b \ : \ b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)] \geq 1/2 + \epsilon(\lambda)$
  $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Good\text{-}Decoder}]$.

- $\mathsf{Cor\text{-}Tr}$: $S_D \subseteq S \ \wedge \ S_D \neq \emptyset$
  $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Cor\text{-}Tr}]$.

- $\mathsf{Fal\text{-}Tr}$: $S_D \setminus S \neq \emptyset$
  $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Fal\text{-}Tr}]$.

A private key traitor tracing scheme $\mathcal{T}$ is said to be f-risky secure if for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot)$, $p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exist negligible functions $\text{negl}_1(\cdot)$, $\text{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \leq \text{negl}_1(\lambda)$ and $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) \cdot f(\lambda, n(\lambda)) - \text{negl}_2(\lambda)$.

Experiment $\mathsf{Expt\text{-}TT\text{-}priv}^{\mathcal{T}}_{\mathcal{A},n,\epsilon}(\lambda)$

- $\big(\mathsf{msk}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_{n(\lambda)})\big) \leftarrow \mathsf{Setup}(1^\lambda, 1^{n(\lambda)})$.
- $(D, m_0, m_1) \leftarrow A^{O_1(\cdot), O_2(\cdot)}()$
- $S_D \leftarrow \mathsf{Trace}^D(\mathsf{msk}, 1^{1/\epsilon(\lambda)}, m_0, m_1)$.

Here, $O_1(\cdot)$ is an oracle that has $\{\mathsf{sk}_1, \mathsf{sk}_2, \ldots, \mathsf{sk}_{n(\lambda)}\}$ hardwired, takes as input an index $i \in \{1, 2, \ldots, n(\lambda)\}$ and outputs sk_i. Let S be the set of indices queried by $\mathcal{A}$.
The oracle $O_2(\cdot)$ is the encryption oracle that has msk hardwired, takes a message m as input, and outputs $\mathsf{Enc}(\mathsf{msk}, m)$.

Figure 2: Experiment $\mathsf{Expt\text{-}TT\text{-}priv}$

Definition 3.7 (Private Key No-Query f-risky secure traitor tracing)**.** Let $\mathcal{T} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$ be a private key traitor tracing scheme. For any polynomial $n(\cdot)$, non-negligible function $\epsilon(\cdot)$ and PPT adversary $\mathcal{A}$, consider the experiment $\mathsf{Expt\text{-}TT\text{-}priv}^{\mathcal{T}}_{\mathcal{A},n,\epsilon}(\lambda)$ (described in Figure 2), except that the adversary $\mathcal{A}$ does not have access to the encryption oracle O_2. Based on this experiment, we can define the following (probabilistic) events $\mathsf{Good\text{-}Decoder}$, $\mathsf{Cor\text{-}Tr}$, $\mathsf{Fal\text{-}Tr}$ and the corresponding probabilities $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}$, $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}$, $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}$ respectively.

A private key traitor tracing scheme $\mathcal{T}$ is said to be no-query f-risky secure if for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot)$, $p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exist negligible functions $\text{negl}_1(\cdot)$, $\text{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \leq \text{negl}_1(\lambda)$ and $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) \cdot f(\lambda, n(\lambda)) - \text{negl}_2(\lambda)$.

4 A New Abstraction for Constructing Risky Traitor Tracing

Let $\{\mathcal{M}_\lambda\}_\lambda$ denote the message space. A mixed bit matching encryption scheme for $\mathcal{M}$ consists of five algorithms with the following syntax.

$\mathsf{Setup}(1^\lambda, 1^\ell) \rightarrow (\mathsf{pk}, \mathsf{msk})$: The setup algorithm takes as input security parameter λ, a parameter ℓ and outputs a public key pk and master secret key msk.

$\mathsf{KeyGen}(\mathsf{msk}, \mathbf{x} \in \{0,1\}^\ell) \rightarrow \mathsf{sk}$: The key generation algorithm takes as input the master secret key msk and a vector $\mathbf{x} \in \{0,1\}^\ell$. It outputs a secret key sk corresponding to $\mathbf{x}$.

$\mathsf{Enc\text{-}PK}(\mathsf{pk}, m \in \mathcal{M}) \rightarrow \mathsf{ct}$: The public-key encryption algorithm takes as input a public key pk and a message m, and outputs a ciphertext ct.

$\mathsf{Enc\text{-}SK}(\mathsf{msk}, m \in \mathcal{M}, \mathbf{y} \in \{0,1\}^\ell) \rightarrow \mathsf{ct}$: The secret-key encryption algorithm takes as input master secret key msk, message m, and an attribute vector $\mathbf{y} \in \{0,1\}^\ell$. It outputs a ciphertext ct.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to z$: The decryption algorithm takes as input a ciphertext ct, a secret key sk and outputs $z \in \mathcal{M} \cup \{\perp\}$.

Permissions Define $f : \{0,1\}^{\ell} \times \{0,1\}^{\ell} \to \{0,1\}$ by the following:

$$f(\mathbf{x}, \mathbf{y}) = \bigwedge_{i=1}^{\ell} x_i \vee y_i$$

We will use this function to determine when secret keys with attribute vectors $\mathbf{x}$ are "permitted" to decrypt ciphertexts with attribute vectors $\mathbf{y}$.

Correctness We require the following properties for correctness:

- For every $\lambda \in \mathbb{N}, \ell \in \mathbb{N}$, $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^{\lambda}, 1^{\ell})$, $\mathbf{x} \in \{0,1\}^{\ell}$, $\mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathbf{x})$, message $m \in \mathcal{M}_{\lambda}$ and $\mathsf{ct} \leftarrow \mathsf{Enc\text{-}PK}(\mathsf{pk}, m)$, $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = m$.
- For every $\lambda \in \mathbb{N}, \ell \in \mathbb{N}$, $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^{\lambda}, 1^{\ell})$, $\mathbf{x} \in \{0,1\}^{\ell}$, $\mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathbf{x})$, message $m \in \mathcal{M}_{\lambda}$, $\mathbf{y} \in \{0,1\}^{\ell}$ and $\mathsf{ct} \leftarrow \mathsf{Enc\text{-}SK}(\mathsf{msk}, m, \mathbf{y})$, if $f(\mathbf{x}, \mathbf{y}) = 1$ then $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = m$.

4.1 Security

Oracles To begin, we define two oracles we use to enable the adversary to query for ciphertexts and secret keys. Let m be a message, and $\mathbf{x}, \mathbf{y}, \in \{0,1\}^{\ell}$.

- $\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}(\mathbf{x}) \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathbf{x})$.
- $\mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}(m, \mathbf{y}) \leftarrow \mathsf{Enc\text{-}SK}(\mathsf{msk}, m, \mathbf{y})$.

Experiments We will now define three security properties that a mixed bit matching encryption scheme must satisfy. These definitions are similar to the indistinguishability-based data/function privacy definitions for attribute based encryption. For each of these experiments we restrict the adversary's queries to the ciphertext and secret key oracles to prevent trivial distinguishing strategies. Also, we will be considering selective definitions, since our constructions achieve selective security, and selective security suffices for our risky traitor tracing application. One could also consider full (adaptive) versions of these security definitions.

Definition 4.1. A mixed bit matching encryption scheme $\mathsf{mBME} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc\text{-}PK}, \mathsf{Enc\text{-}SK}, \mathsf{Dec})$ is said to satisfy *pk-sk ciphertext indistinguishability* if for any polynomial $\ell(\cdot)$ and stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all security parameters $\lambda \in \mathbb{N}$, $\Pr[1 \leftarrow \mathsf{Expt\text{-}pk\text{-}sk\text{-}ct}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^{\lambda})] \leq 1/2 + \mathrm{negl}(\lambda)$, where $\mathsf{Expt\text{-}pk\text{-}sk\text{-}ct}$ is defined in Figure 3.

Experiment $\mathsf{Expt\text{-}pk\text{-}sk\text{-}ct}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^{\lambda})$

- $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^{\lambda}, 1^{\ell(\lambda)})$
- $m \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{pk})$.
- $\mathsf{ct}_0 \leftarrow \mathsf{Enc\text{-}SK}(\mathsf{msk}, m, 1^{\ell(\lambda)})$, $\mathsf{ct}_1 \leftarrow \mathsf{Enc\text{-}PK}(\mathsf{pk}, m)$, $b \leftarrow \{0,1\}$
- $b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{ct}_b)$.
- Output 1 if $b = b'$, and 0 otherwise.

Figure 3: Public-key vs Secret-key Ciphertext Indistinguishability Experiment

Definition 4.2. A mixed bit matching encryption scheme $\mathsf{mBME} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc\text{-}PK}, \mathsf{Enc\text{-}SK}, \mathsf{Dec})$ is said to satisfy *selective ciphertext hiding* if for any polynomial $\ell(\cdot)$ and stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all security parameters $\lambda \in \mathbb{N}$, $\Pr[1 \leftarrow \mathsf{Expt\text{-}ct\text{-}ind}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^\lambda)] \leq 1/2 + \mathrm{negl}(\lambda)$, where $\mathsf{Expt\text{-}ct\text{-}ind}$ is defined in Figure 4.

Experiment $\mathsf{Expt\text{-}ct\text{-}ind}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^\lambda)$

- $(\mathbf{y}_0, \mathbf{y}_1) \leftarrow \mathcal{A}(1^\lambda)$.
- $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell(\lambda)})$
- $(m_0, m_1) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{pk})$
- $b \leftarrow \{0,1\}$, $\mathsf{ct}_b \leftarrow \mathsf{Enc\text{-}SK}(\mathsf{msk}, m_b, \mathbf{y}_b)$
- $b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{ct}_b)$
- Output 1 if $b = b'$, and 0 otherwise.

Adversarial Restrictions: For all queries $\mathbf{x}$ made by $\mathcal{A}$ to $\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}$ the following conditions must hold:

- If $m_0 = m_1$, then $f(\mathbf{x}, \mathbf{y}_0) = f(\mathbf{x}, \mathbf{y}_1)$.
- If $m_0 \neq m_1$, then $f(\mathbf{x}, \mathbf{y}_0) = f(\mathbf{x}, \mathbf{y}_1) = 0$.

Figure 4: Ciphertext Hiding Experiment

Definition 4.3. A mixed bit matching encryption scheme $\mathsf{mBME} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc\text{-}PK}, \mathsf{Enc\text{-}SK}, \mathsf{Dec})$ is said to satisfy *selective key hiding* if for any polynomial $\ell(\cdot)$ and stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all security parameters $\lambda \in \mathbb{N}$, $\Pr[1 \leftarrow \mathsf{Expt\text{-}key\text{-}ind}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^\lambda)] \leq 1/2 + \mathrm{negl}(\lambda)$, where $\mathsf{Expt\text{-}key\text{-}ind}$ is defined in Figure 5.

Experiment $\mathsf{Expt\text{-}key\text{-}ind}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^\lambda)$

- $(\mathbf{x}_0, \mathbf{x}_1) \leftarrow \mathcal{A}(1^\lambda)$
- $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell(\lambda)})$
- $b \leftarrow \{0,1\}$, $\mathsf{sk}_b \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathbf{x}_b)$
- $b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{pk}, \mathsf{sk}_b)$
- Output 1 if $b = b'$, and 0 otherwise.

Adversarial Restrictions: For all queries $(m, \mathbf{y})$ made by $\mathcal{A}$ to $\mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}$ the following equality must hold: $f(\mathbf{x_0}, \mathbf{y}) = f(\mathbf{x_1}, \mathbf{y})$.

Figure 5: Key Hiding Experiment

4.2 Simplified Ciphertext Hiding

As a tool for proving mixed bit matching encryption constructions secure, we define two simplified ciphertext hiding experiments, and then show that they imply the original (selective) ciphertext hiding security game.

Definition 4.4. A mixed bit matching encryption scheme $\mathsf{mBME} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc\text{-}PK}, \mathsf{Enc\text{-}SK}, \mathsf{Dec})$ is said to satisfy *selective 1-attribute ciphertext hiding* if for any polynomial $\ell(\cdot)$ and stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all security parameters $\lambda \in \mathbb{N}$, $\Pr[1 \leftarrow \mathsf{Expt\text{-}1\text{-}attr\text{-}ct\text{-}ind}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^\lambda)] \leq 1/2 + \mathrm{negl}(\lambda)$, where $\mathsf{Expt\text{-}1\text{-}attr\text{-}ct\text{-}ind}$ is defined in Figure 6.

Experiment $\mathsf{Expt\text{-}1\text{-}attr\text{-}ct\text{-}ind}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^\lambda)$

- $(\mathbf{y}_0, \mathbf{y}_1) \leftarrow \mathcal{A}(1^\lambda)$ where $y_{0,i} \neq y_{1,i}$ for at most one $i \in \{1, \ldots, \ell(\lambda)\}$.
- $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell(\lambda)})$
- $m \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{pk})$
- $b \leftarrow \{0,1\}$, $\mathsf{ct}_b \leftarrow \mathsf{Enc\text{-}SK}(\mathsf{msk}, m_b, \mathbf{y}_b)$.
- $b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{pk}, \mathsf{ct}_b)$
- Output 1 if $b = b'$, and 0 otherwise.

Adversarial Restrictions: For all queries $\mathbf{x}$ made by $\mathcal{A}$ to $\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}$ the following equality must hold: $f(\mathbf{x}, \mathbf{y}_0) = f(\mathbf{x}, \mathbf{y}_1)$.

Figure 6: 1-Attribute Ciphertext Hiding Experiment

Definition 4.5. A mixed bit matching encryption scheme $\mathsf{mBME} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc\text{-}PK}, \mathsf{Enc\text{-}SK}, \mathsf{Dec})$ is said to satisfy *selective ciphertext indistinguishability under chosen attributes* if for any polynomial $\ell(\cdot)$ and stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all security parameters $\lambda \in \mathbb{N}$, $\Pr[1 \leftarrow \mathsf{Expt\text{-}IND\text{-}CA}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^\lambda)] \leq 1/2 + \mathrm{negl}(\lambda)$, where $\mathsf{Expt\text{-}IND\text{-}CA}$ is defined in Figure 7.

Experiment $\mathsf{Expt\text{-}IND\text{-}CA}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^\lambda)$

- $\mathbf{y} \leftarrow \mathcal{A}(1^\lambda)$.
- $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell(\lambda)})$
- $(m_0, m_1) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{pk})$
- $b \leftarrow \{0,1\}$, $\mathsf{ct}_b \leftarrow \mathsf{Enc\text{-}SK}(\mathsf{msk}, m_b, \mathbf{y})$.
- $b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{pk}, \mathsf{ct}_b)$
- Output 1 if $b = b'$, and 0 otherwise.

Adversarial Restrictions: If $m_0 \neq m_1$, then for all queries $\mathbf{x}$ made by $\mathcal{A}$ to $\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}$ the following equality must hold: $f(\mathbf{x}, \mathbf{y}) = 0$.

Figure 7: Ciphertext Indistinguishability under Chosen Attributes Experiment

Theorem 4.1. If a mixed bit matching encryption scheme $\mathsf{mBME} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc\text{-}PK}, \mathsf{Enc\text{-}SK}, \mathsf{Dec})$ satisfies selective 1-attribute ciphertext hiding (Definition 4.4) and selective ciphertext indistinguishability under chosen attributes (Definition 4.5), then it also satisfies selective ciphertext hiding (Definition 4.2).

The proof of above theorem is provided later in Appendix A.

4.3 Simplified Key Hiding

We also define a similar simplified experiment for the key hiding security property.

Definition 4.6. A mixed bit matching encryption scheme $\mathsf{mBME} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc\text{-}PK}, \mathsf{Enc\text{-}SK}, \mathsf{Dec})$ is said to satisfy *selective 1-attribute key hiding* if for any polynomial ℓ and stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all security parameters $\lambda \in \mathbb{N}$, $\Pr[1 \leftarrow \mathsf{Expt\text{-}1\text{-}attr\text{-}key\text{-}ind}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^\lambda)] \leq 1/2 + \mathrm{negl}(\lambda)$, where $\mathsf{Expt\text{-}1\text{-}attr\text{-}key\text{-}ind}$ is defined in Figure 8.

Theorem 4.2. If a mixed bit matching encryption scheme $\mathsf{mBME} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc\text{-}PK}, \mathsf{Enc\text{-}SK}, \mathsf{Dec})$ satisfies 1-attribute key hiding (Definition 4.6) then it satisfies key hiding (Definition 4.3).

The proof of above theorem is provided later in Appendix A.

Experiment $\mathsf{Expt\text{-}1\text{-}attr\text{-}key\text{-}ind}^{\mathsf{mBME}}_{\ell(\lambda),\mathcal{A}}(1^\lambda)$

- $(\mathbf{x}_0, \mathbf{x}_1) \leftarrow \mathcal{A}(1^\lambda)$ where $x_{0,i} \neq x_{1,i}$ for at most one i.
- $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell(\lambda)})$.
- $b \leftarrow \{0,1\}$, $\mathsf{sk}_b \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathbf{x}_b)$.
- $b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{pk}, \mathsf{sk}_b)$.
- Output 1 if $b = b'$.

Adversarial Restrictions: For all queries $(m, \mathbf{y})$ made by $\mathcal{A}$ to $\mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}$ the following equality must hold: $f(\mathbf{x}_0, \mathbf{y}) = f(\mathbf{x}_1, \mathbf{y})$.

Figure 8: 1-Attribute Key Hiding Experiment

5 Building Risky Traitor Tracing using Mixed Bit Matching Encryption

In this section, we provide a generic construction for risky traitor tracing schemes from any mixed bit matching encryption scheme. Our transformation leads to a risky traitor tracing scheme with secret-key tracing. The *risky-ness* of the scheme will be $f = \frac{k}{n+k-1} - O\left(\frac{k(k-1)}{n^2}\right)$,[6] where k can be thought of as a scheme parameter fixed during setup, and the size of ciphertext will grow with k.

5.1 Construction

- $\mathsf{Setup}(1^\lambda, 1^n)$: The setup algorithm chooses a key pair for mixed bit matching encryption system as $(\mathsf{mbme.pk}, \mathsf{mbme.msk}) \leftarrow \mathsf{mBME.Setup}(1^\lambda, 1^{k+1})$. Next, it samples an index w as $w \leftarrow \{-k+2, -k+3, \ldots, n-1, n\}$, and sets vectors $\mathbf{x}_i$ for $i \in [n]$ as

$$\mathbf{x}_i = \begin{cases} 0^{k+1} & \text{if } i < w, \\ 0^{k-i+w}1^{i-w+1} & \text{if } w \leq i < w+k, \\ 1^{k+1} & \text{otherwise.} \end{cases}$$

  It sets the master secret key as $\mathsf{msk} = (\mathsf{mbme.msk}, w)$, public key as $\mathsf{pk} = \mathsf{mbme.pk}$, and computes the n user secret keys as $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{mbme.msk}, \mathbf{x}_i)$ for $i \in [n]$.

- $\mathsf{Enc}(\mathsf{pk}, m)$: The encryption algorithm outputs the ciphertext ct as $\mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}PK}(\mathsf{pk}, m)$.

- $\mathsf{Dec}(\mathsf{sk}, m)$: The decryption algorithm outputs the message m as $m = \mathsf{mBME.Dec}(\mathsf{sk}, \mathsf{ct})$.

- $\mathsf{Trace}^D(\mathsf{msk}, 1^y, m_0, m_1)$: Let $\mathsf{msk} = (\mathsf{mbme.msk}, w)$. To define the trace algorithm, we first define a special index encryption algorithm $\mathsf{Enc\text{-}ind}$ which takes as input a master secret key msk, message m, and an index $i \in [k+1]$.

  $\mathsf{Enc\text{-}ind}(\mathsf{msk}, m, i)$: The index encryption algorithm outputs $\mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}SK}(\mathsf{msk}, m, 1^{k+1-i}0^i)$.

  Next, consider the $\mathsf{Subtrace}$ algorithm defined in Figure 9. The sub-tracing algorithm simply tests whether the decoder box uses the key for user $i + w - 1$ where i is one of the inputs provided to $\mathsf{Subtrace}$. Now the tracing algorithm simply runs the $\mathsf{Subtrace}$ algorithm for all indices $i \in [k]$, and for each index i where the $\mathsf{Subtrace}$ algorithm outputs 1, the tracing algorithm adds index $i + w - 1$ to the set of traitors. Concretely, the algorithm runs as follows:

  - Let $S = \emptyset$. For $i = 1$ to k:

[6] We want to point out that for $k = 1$ we get the tight *risky-ness*, i.e. prove that our scheme is $\frac{1}{n}$-risky secure.

    * Compute $b \leftarrow \mathsf{Subtrace}(\mathsf{mbme.msk}, 1^y, m_0, m_1, i)$.
    * If $b = 1$, set $S := S \cup \{i + w - 1\}$.
  - Output S.

Algorithm $\mathsf{Subtrace}(\mathsf{msk}, 1^y, m_0, m_1, i)$

Inputs: Key msk, parameter y, messages m_0, m_1, index i
Output: 0/1
Let $\epsilon = \lfloor 1/y \rfloor$. It sets $T = \lambda \cdot n/\epsilon$, and $\mathsf{count}_1 = \mathsf{count}_2 = 0$. For $j = 1$ to T, it computes the following:

1. It chooses $b_j \leftarrow \{0,1\}$ and computes $\mathsf{ct}_{j,1} \leftarrow \mathsf{Enc\text{-}ind}(\mathsf{msk}, m_{b_j}, i)$ and sends $\mathsf{ct}_{j,1}$ to D. If D outputs b_j, set $\mathsf{count}_1 = \mathsf{count}_1 + 1$, else set $\mathsf{count}_1 = \mathsf{count}_1 - 1$.
2. It chooses $c_j \leftarrow \{0,1\}$ and computes $\mathsf{ct}_{j,2} \leftarrow \mathsf{Enc\text{-}ind}(\mathsf{msk}, m_{c_j}, i{+}1)$ and sends $\mathsf{ct}_{j,2}$ to D. If D outputs c_j, set $\mathsf{count}_2 = \mathsf{count}_2 + 1$, else set $\mathsf{count}_2 = \mathsf{count}_2 - 1$.

If $\mathsf{count}_1 - \mathsf{count}_2 > T \cdot (\epsilon/4n)$, output 1, else output 0.

Figure 9: Subtrace

Correctness. Since the encryption algorithm simply runs the public-key encryption algorithm for mixed bit matching encryption the correctness of above scheme follows directly from the correctness of the mixed bit matching encryption scheme.

Singular Trace Property. Note that if k is fixed to be 1, then our scheme satisfies the singular trace property as defined in Definition 3.5. This is because the trace algorithm will either output the fixed index w (chosen during setup), or output an empty set.

5.2 Proof of Security

5.2.1 IND-CPA Security

First, we will show that the above traitor tracing scheme is IND-CPA secure. Formally, we prove the following.

Theorem 5.1. If the mixed bit matching encryption scheme mBME = ($\mathsf{mBME.Setup}$, $\mathsf{mBME.KeyGen}$, $\mathsf{mBME.Enc\text{-}PK}$, $\mathsf{mBME.Enc\text{-}SK}$, $\mathsf{mBME.Dec}$) satisfies *pk-sk ciphertext indistinguishability* and *ciphertext hiding* properties (Definitions 4.1 and 4.2), then the traitor tracing scheme described in Section 5.1 is IND-CPA secure (Definition 3.1).

Proof. Our proof proceeds via a sequence of hybrid games. Each game is played between the IND-CPA challenger and attacker $\mathcal{A}$. Let $\mathcal{A}$ be any PPT adversary that wins the IND-CPA game with non-negligible advantage. We argue that such an adversary must break either pk-sk ciphertext indistinguishability, or ciphertext hiding security of the underlying mixed bit matching encryption scheme. The first game corresponds to the IND-CPA game as described in Definition 3.1.

We will first define the sequence of hybrid games, and then show that they are computationally indistinguishable.

Game 1: This corresponds to the original IND-CPA game.

- **Setup Phase.** The challenger runs the Setup algorithm to generate the public-secret keys $(\mathsf{msk}, \mathsf{pk}, \{\mathsf{sk}_i\}_i)$. It sends the public key pk to the attacker $\mathcal{A}$.
- **Challenge Phase.** $\mathcal{A}$ sends two messages m_0, m_1 to the challenger. The challenger chooses a random bit $b \leftarrow \{0,1\}$ and computes the challenge ciphertext as $\mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}PK}(\mathsf{pk}, m_b)$. It sends ct to the attacker.
- **Guess.** $\mathcal{A}$ outputs its guess b', and wins if $b = b'$.

$\mathsf{Game}\ 2$: This is identical to the previous game, except the challenger computes the challenge ciphertext ct as a secret-key ciphertext instead of a public-key ciphertext.

- **Challenge Phase.** $\mathcal{A}$ sends two messages m_0, m_1 to the challenger. The challenger chooses a random bit $b \leftarrow \{0,1\}$ and computes the challenge ciphertext as $\mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}SK}(\mathsf{msk}, m_b, 1^{k+1})$. It sends ct to the attacker.

Let $\mathsf{Adv}^i_{\mathcal{A}} = |\Pr[b' = b] - 1/2|$ denote the advantage of adversary $\mathcal{A}$ in guessing the bit b in Game i. To complete the proof, we establish via a sequence of lemmas that no PPT adversary $\mathcal{A}$ can distinguish between $\mathsf{Game}\ 1$ and 2 with non-negligible probability, as well as can have non-negligible advantage in $\mathsf{Game}\ 2$. Below we discuss our lemmas in detail.

Lemma 5.1. If $\mathsf{mBME} = (\mathsf{mBME.Setup}, \mathsf{mBME.KeyGen}, \mathsf{mBME.Enc\text{-}PK}, \mathsf{mBME.Enc\text{-}SK}, \mathsf{mBME.Dec})$ satisfies *pk-sk ciphertext indistinguishability*, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}|$ is negligible in the security parameter λ.

Proof. Suppose there exists an adversary $\mathcal{A}$ such that $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}|$ is non-negligible. We construct an algorithm $\mathcal{B}$ that can break pk-sk ciphertext indistinguishability property of the underlying mixed bit matching encryption scheme.

The mixed bit matching encryption scheme challenger samples mBME key pair $(\mathsf{pk}, \mathsf{msk})$, and sends pk to the reduction algorithm $\mathcal{B}$ which it forwards to adversary $\mathcal{A}$. Next, $\mathcal{A}$ sends two challenge messages m_0, m_1 to the reduction algorithm $\mathcal{B}$. $\mathcal{B}$ then chooses a random bit $b \leftarrow \{0,1\}$, and sends m_b to the mBME challenger as its challenge message. The challenger replies with challenge ciphertext ct which $\mathcal{B}$ forwards to $\mathcal{A}$ as its challenge. Finally, $\mathcal{A}$ outputs its guess b'. If $b = b'$, then $\mathcal{B}$ sends 0 as its guess (i.e., ct was public-key ciphertext), otherwise it sends 1 as its guess (i.e., ct was secret-key ciphertext) as its guess to the mBME challenger.

Note that if the mBME challenger computed ct as $\mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}PK}(\mathsf{pk}, m_b)$, then $\mathcal{B}$ perfectly simulates Game 1 for adversary $\mathcal{A}$. Otherwise it simulates Game 2 for $\mathcal{A}$. As a result, if $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}|$ is non-negligible, then $\mathcal{B}$ breaks pk-sk ciphertext indistinguishability with non-negligible advantage. ■

Lemma 5.2. If $\mathsf{mBME} = (\mathsf{mBME.Setup}, \mathsf{mBME.KeyGen}, \mathsf{mBME.Enc\text{-}PK}, \mathsf{mBME.Enc\text{-}SK}, \mathsf{mBME.Dec})$ satisfies *ciphertext hiding*, then for all PPT adversaries $\mathcal{A}$, $\mathsf{Adv}^2_{\mathcal{A}}$ is negligible in the security parameter λ.

Proof. Suppose there exists an adversary $\mathcal{A}$ such that $\mathsf{Adv}^2_{\mathcal{A}}$ is non-negligible. We construct an algorithm $\mathcal{B}$ that can break ciphertext hiding property of the underlying mixed bit matching encryption scheme.

The mixed bit matching encryption scheme challenger samples mBME key pair $(\mathsf{pk}, \mathsf{msk})$, and sends pk to the reduction algorithm $\mathcal{B}$ which it forwards to adversary $\mathcal{A}$. Next, $\mathcal{A}$ sends two challenge messages m_0, m_1 to the reduction algorithm $\mathcal{B}$. $\mathcal{B}$ then sends $((m_0, 1^{k+1}), (m_1, 1^{k+1}))$ to the mBME challenger as its challenge message-attribute pairs. The challenger replies with challenge ciphertext ct which $\mathcal{B}$ forwards to $\mathcal{A}$ as its challenge. Finally, $\mathcal{A}$ outputs its guess b', which $\mathcal{B}$ forwards to the mBME challenger as its guess.

Note that $\mathcal{B}$ perfectly simulates Game 2 for adversary $\mathcal{A}$. Therefore, if $\mathsf{Adv}^2_{\mathcal{A}}$ is non-negligible, then $\mathcal{B}$ breaks ciphertext hiding with non-negligible advantage. ■

■

5.2.2 False Trace Probability

Next, we will show that an honest party will not be implicated by our trace algorithm with non-negligible probability.

Theorem 5.2. If $\mathsf{mBME} = (\mathsf{mBME.Setup}, \mathsf{mBME.KeyGen}, \mathsf{mBME.Enc\text{-}PK}, \mathsf{mBME.Enc\text{-}SK}, \mathsf{mBME.Dec})$ satisfies *ciphertext hiding* (Definition 4.2), then for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), k(\cdot)$, $p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$,

$$\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \leq \mathrm{negl}(\lambda),$$

where $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\cdot)$ is as defined in Definition 3.4.

Proof. Given any pirate decoder box D, let $p_D^{(\mathsf{ind})} = \Pr[D(\mathsf{ct}) = b : \mathsf{ct} \leftarrow \mathsf{Enc\text{-}ind}(\mathsf{msk}, m_b, \mathsf{ind})]$ for $\mathsf{ind} \in [k{+}1]$, where the probability is taken over bit b, random coins of decoder D and randomness used during encryption. Let $\mathsf{Diff\text{-}Adv}_{\mathsf{ind}}$ denote the event when the advantage of decoder D in distinguishing index-ind encryptions of m_0 and m_1 is $\epsilon/8n$ more than its advantage in distinguishing index-$(\mathsf{ind}+1)$ encryptions. Formally, for $\mathsf{ind} \in [k]$, let

$$\mathsf{Diff\text{-}Adv}_{\mathsf{ind}} \;:\; p_D^{(\mathsf{ind})} - p_D^{(\mathsf{ind}+1)} > \epsilon/8n.$$

Also, for $\mathsf{ind} \in [k]$, let $\mathsf{Fal\text{-}SubTr}_{\mathsf{ind}}$ denote the event $\mathsf{Fal\text{-}Tr} \wedge (\mathsf{ind} \in S_D)$ where S_D corresponds the set of traitors output by the Trace algorithm. In other words, $\mathsf{Fal\text{-}SubTr}_{\mathsf{ind}}$ occurs when ind^{th} $\mathsf{Subtrace}$ outputs 1 and $\mathcal{A}$ had not queried for the corresponding secret key.

First, note that we could rewrite the probability of a *false trace* as

$$\begin{aligned}\Pr[\mathsf{Fal\text{-}Tr}] &\leq \sum_{\mathsf{ind}\in[k]} \Pr[\mathsf{Fal\text{-}SubTr}_{\mathsf{ind}}] \\ &\leq \sum_{\mathsf{ind}\in[k]} \Pr[\mathsf{Fal\text{-}SubTr}_{\mathsf{ind}} \mid \overline{\mathsf{Diff\text{-}Adv}_{\mathsf{ind}}}] + \sum_{\mathsf{ind}\in[k]} \Pr[\mathsf{Fal\text{-}SubTr}_{\mathsf{ind}} \wedge \mathsf{Diff\text{-}Adv}_{\mathsf{ind}}].\end{aligned}$$

To bound the overall probability of a false trace, we start by showing that $\Pr[\mathsf{Fal\text{-}SubTr}_{\mathsf{ind}} \mid \overline{\mathsf{Diff\text{-}Adv}_{\mathsf{ind}}}] \leq \mathrm{negl}_1(\lambda)$ by using a simple Chernoff bound. Next, we show that $\Pr[\mathsf{Fal\text{-}SubTr}_{\mathsf{ind}} \wedge \mathsf{Diff\text{-}Adv}_{\mathsf{ind}}] \leq \mathrm{negl}_2(\lambda)$ by relying on *ciphertext hiding* property of the mixed bit matching encryption scheme. These two lemmas together imply that $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda)$ is also bounded by a negligible function.

Lemma 5.3. For every adversary $\mathcal{A}$, there exists a negligible function $\mathrm{negl}_1(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\mathsf{ind} \in [k]$

$$\Pr[\mathsf{Fal\text{-}SubTr}_{\mathsf{ind}} \mid \overline{\mathsf{Diff\text{-}Adv}_{\mathsf{ind}}}] \leq \mathrm{negl}_1(\lambda).$$

Proof. Consider the binary random variables $X^{(\mathsf{ind})}$ for $\mathsf{ind} \in [k+1]$ defined as

$$X^{(\mathsf{ind})} = \begin{cases} 1 & \text{with probability } p_D^{(\mathsf{ind})}, \\ 0 & \text{otherwise.}\end{cases}$$

Let $Z^{(\mathsf{ind})}$ be another random variable defined as $Z^{(\mathsf{ind})} = X^{(\mathsf{ind})} - X^{(\mathsf{ind}+1)}$. Now using linearity of expectation, we can write that

$$\mathbb{E}[Z^{(\mathsf{ind})} \mid \overline{\mathsf{Diff\text{-}Adv}_{\mathsf{ind}}}] \leq \epsilon/8n.$$

Also, we know that the sub-tracing algorithm estimates $\mathbb{E}[Z^{(\mathsf{ind})}]$ by independently sampling $T = \lambda \cdot n/\epsilon$ elements from the distribution induced by $Z^{(\mathsf{ind})}$. In other words, in each trial it first computes a single index-ind and index-$(\mathsf{ind}+1)$ encryptions of messages m_0 and m_1 using uniform randomness, then it uses the decoder box D to decrypt each ciphertext and sets the value of sampled variable appropriately. Let $z_i^{(\mathsf{ind})}$ be the sampled value in i^{th} trial. Now we know that $\mathsf{count}_1^{(\mathsf{ind})} - \mathsf{count}_2^{(\mathsf{ind})} = \sum_{i=1}^{T} z_i^{(\mathsf{ind})}$. Thus, we can write that

$$\Pr[\mathsf{Fal\text{-}SubTr}_{\mathsf{ind}} \mid \overline{\mathsf{Diff\text{-}Adv}_{\mathsf{ind}}}] = \Pr\left[\sum_{i=1}^{T} z_i^{(\mathsf{ind})} > T \cdot \epsilon/4n \mid \overline{\mathsf{Diff\text{-}Adv}_{\mathsf{ind}}}\right].$$

Using a Chernoff bound, we can bound the above probability as

$$\Pr[\mathsf{Fal\text{-}SubTr}_{\mathsf{ind}} \mid \overline{\mathsf{Diff\text{-}Adv}_{\mathsf{ind}}}] \leq e^{-T\epsilon/16n}.$$

Substituting $T = \lambda \cdot n/\epsilon$, we get $\Pr[\mathsf{Fal\text{-}SubTr_{ind}} \mid \overline{\mathsf{Diff\text{-}Adv_{ind}}}] \leq 2^{-O(\lambda)} = \mathrm{negl}_1(\lambda)$. This completes the proof. ∎

Lemma 5.4. If mBME satisfies *ciphertext hiding*, then for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, and every $\mathsf{ind} \in [k]$,

$$\Pr[\mathsf{Fal\text{-}SubTr_{ind}} \wedge \mathsf{Diff\text{-}Adv_{ind}}] \leq \mathrm{negl}_2(\lambda).$$

Here, the events $\mathsf{Fal\text{-}SubTr_{ind}}$ and $\mathsf{Diff\text{-}Adv_{ind}}$ are parameterized by the adversary $\mathcal{A}$.

Proof. The proof of above lemma is independent of the choice of index ind. Thus, for the purpose of this proof, consider ind as a fixed value. Note that it is sufficient to show that $\Pr[\mathsf{Fal\text{-}SubTr_{ind}} \wedge \mathsf{Diff\text{-}Adv_{ind}}]$ is bounded by a negligible function. Suppose, on the contrary, there exists a PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible functions $\epsilon(\cdot), \delta(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/q(\lambda)$, $\Pr[\mathsf{Fal\text{-}SubTr_{ind}} \wedge \mathsf{Diff\text{-}Adv_{ind}}] \geq \delta(\lambda)$. Then we can construct an algorithm $\mathcal{B}$ that can break ciphertext hiding property of the underlying mixed bit matching encryption scheme.

The reduction algorithm $\mathcal{B}$ chooses an index w as $w \leftarrow \{-k+2, \ldots, n(\lambda)\}$, and sends the challenge attribute pair as $(\mathbf{y}_0, \mathbf{y}_1) = (1^{k+1-\mathsf{ind}}0^{\mathsf{ind}}, 1^{k-\mathsf{ind}}0^{\mathsf{ind}+1})$ to the mixed bit matching encryption scheme challenger. The mBME challenger samples mBME key pair $(\mathsf{pk}, \mathsf{msk})$, and sends pk to the reduction algorithm $\mathcal{B}$ which it forwards to adversary $\mathcal{A}$. Next, $\mathcal{A}$ makes key queries to $\mathcal{B}$ which are answered as follows. For each key query $i \in [n(\lambda)]$, if $i = w + \mathsf{ind} - 1$ then $\mathcal{B}$ aborts (and outputs a random bit), otherwise $\mathcal{B}$ makes secret key query for vector $\mathbf{x}_i$ to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$. (Here $\mathbf{x}_i$ is as defined in the construction. Note that $\mathbf{x}_i$ depends on the index w chosen by $\mathcal{B}$.) After all these queries, the adversary $\mathcal{A}$ sends a decoder box D and messages m_0, m_1 to $\mathcal{B}$.

$\mathcal{B}$ then chooses two bits α, β uniformly at random, i.e. $\alpha, \beta \leftarrow \{0,1\}$. Next, $\mathcal{B}$ sends messages (m_α, m_α) as its challenge messages (i.e., a single challenge message m_α). It receives challenge ciphertext ct^* from mBME challenger. $\mathcal{B}$ then also queries the mBME challenger for a (secret-key) encryption of m_α for vector $\mathbf{y}_\beta$. Let ct be the challenger's response. Finally, $\mathcal{B}$ runs decoder box D on ct and ct^* independently, and if $D(\mathsf{ct}) = D(\mathsf{ct}^*)$, it outputs $b' = \beta$, else it outputs $b' = 1 - \beta$ as its guess.

First, note that $\mathcal{B}$ does not make any secret key query for vector $\mathbf{x}_{w+\mathsf{ind}-1} = 0^{k-\mathsf{ind}+1}1^{\mathsf{ind}}$, but only makes key queries for vectors $0^{k-j+1}1^j$ for $j \neq \mathsf{ind}$. (Recall it aborts if $i = w + \mathsf{ind} - 1$.) Since all these queries are allowed as $f(\mathbf{x}_i, \mathbf{y}_0) = f(\mathbf{x}_i, \mathbf{y}_1)$ for all $i \neq w + \mathsf{ind} - 1$ (where f denotes the 'AND-of-ORs'), thus $\mathcal{B}$ is an admissible adversary in the key hiding game. Now we analyse $\mathcal{B}$'s advantage in breaking ciphertext hiding security.

Let $p^D_{j,b} = \Pr[D(\mathsf{ct}) = b \ : \ \mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}SK}(\mathsf{msk}, m_b, 1^{k+1-j}0^j)]$, where the probability is taken over the coins of decoder D and encryption algorithm. Recall we have that $\Pr[\mathsf{Fal\text{-}SubTr_{ind}} \wedge \mathsf{Diff\text{-}Adv_{ind}}] \geq \delta(\lambda)$. Therefore, we can write that

$$\Pr\left[\mathsf{Fal\text{-}SubTr_{ind}} \wedge \left((p^D_{\mathsf{ind},0} + p^D_{\mathsf{ind},1})/2 - (p^D_{\mathsf{ind}+1,0} + p^D_{\mathsf{ind}+1,1})/2\right) \geq \epsilon/8n\right] \geq \delta(\lambda).$$

Thus, we can also write that there exists a bit b such that

$$\Pr\left[\mathsf{Fal\text{-}SubTr_{ind}} \wedge \left(p^D_{\mathsf{ind},b} - p^D_{\mathsf{ind}+1,b}\right) \geq \epsilon/8n\right] \geq \delta(\lambda).$$

Now since the reduction algorithm $\mathcal{B}$ simply randomly guesses this bit b, thus we have that

$$\Pr\left[\mathsf{Fal\text{-}SubTr_{ind}} \wedge \left(p^D_{\mathsf{ind},\alpha} - p^D_{\mathsf{ind}+1,\alpha}\right) \geq \epsilon/8n\right] \geq \delta(\lambda)/2.$$

Let β^* denote the challenger's bit (i.e., ct^* encrypts m_α for index $\mathbf{y}_{\beta^*}$). Next, consider the following probability:

$$\rho_{D,m} = \Pr\left[\beta^* = b' : \begin{array}{l} \beta^* \leftarrow \{0,1\}; \mathsf{ct}^* \leftarrow \mathsf{mBME.Enc\text{-}SK}(\mathsf{msk}, m, \mathbf{y}_{\beta^*}); \\ \beta \leftarrow \{0,1\}; \mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}SK}(\mathsf{msk}, m, \mathbf{y}_\beta); \\ b' = \beta \text{ if } D(\mathsf{ct}^*) = D(\mathsf{ct}), \text{ else } b' = 1 - \beta \end{array}\right].$$

Since the decoder D is run on ciphertexts $\mathsf{ct}^*, \mathsf{ct}$ independently, we could rewrite the above probability (for any message m_α) as follows:

$$\begin{aligned}
\rho_{D,m_\alpha} = &\ \Pr\left[D(\mathsf{ct}) = \beta : \beta \leftarrow \{0,1\}; \mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}SK}(\mathsf{msk}, m_\alpha, \mathbf{y}_\beta)\right]^2 \\
&+ \Pr\left[D(\mathsf{ct}) \neq \beta : \beta \leftarrow \{0,1\}; \mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}SK}(\mathsf{msk}, m_\alpha, \mathbf{y}_\beta)\right]^2 \\
= &\left(\frac{1}{2} + (-1)^\alpha \cdot \frac{p^D_{\mathsf{ind},\alpha} - p^D_{\mathsf{ind}+1,\alpha}}{2}\right)^2 + \left(\frac{1}{2} - (-1)^\alpha \cdot \frac{p^D_{\mathsf{ind},\alpha} - p^D_{\mathsf{ind}+1,\alpha}}{2}\right)^2 \\
= &\ \frac{1}{2} + \left(p^D_{\mathsf{ind},\alpha} - p^D_{\mathsf{ind}+1,\alpha}\right)^2.
\end{aligned}$$

Now we know that with probability $\delta(\lambda)/2$, we have that $\mathsf{Fal\text{-}SubTr_{ind}} \wedge \left(p^D_{\mathsf{ind},\alpha} - p^D_{\mathsf{ind}+1,\alpha}\right) \geq \epsilon/8n$ occurs. From above inequality, we get that

$$\Pr\left[\mathcal{B} \text{ wins} \mid \left(\mathsf{Fal\text{-}SubTr_{ind}} \wedge \left(p^D_{\mathsf{ind},\alpha} - p^D_{\mathsf{ind}+1,\alpha}\right) \geq \epsilon/8n\right)\right] \geq \frac{1}{2} + \left(\frac{\epsilon}{8n}\right)^2.$$

Also, since $\left(p^D_{\mathsf{ind},\alpha} - p^D_{\mathsf{ind}+1,\alpha}\right)^2 \geq 0$, thus for any decoder D and message m_α, $\rho_{D,m_\alpha} \geq 1/2$. This implies that

$$\Pr\left[\mathcal{B} \text{ wins} \mid \overline{\left(\mathsf{Fal\text{-}SubTr_{ind}} \wedge \left(p^D_{\mathsf{ind},\alpha} - p^D_{\mathsf{ind}+1,\alpha}\right) \geq \epsilon/8n\right)}\right] \geq \frac{1}{2}.$$

Combining above equations, we get that $\Pr[\mathcal{B} \text{ wins}] \geq \frac{1}{2} + \frac{\delta}{2} \cdot \left(\frac{\epsilon}{8n}\right)^2$. Thus, the lemma follows.

■

■

5.2.3 Correct Trace Probability

We will first introduce some notations for our security proof. For any $\gamma \in [0, 1/2]$, a decoder box D is said to be γ-Dist for messages m_0, m_1 if

$$\Pr\left[D(\mathsf{ct}) = b \ : \ b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)\right] \geq 1/2 + \gamma.$$

Similarly, for any index $0 \leq \mathsf{ind} \leq k+1$, a decoder box D is said to be γ-$\mathsf{Dist}^{(\mathsf{ind})}$ for messages m_0, m_1 if

$$\Pr[D(\mathsf{ct}) = b \ : \ b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc\text{-}ind}(\mathsf{msk}, m_b, \mathsf{ind})] \geq 1/2 + \gamma.$$

For any adversary $\mathcal{A}$ and polynomial $n(\cdot)$, we define experiment $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)$ (see Figure 10). The experiment takes as input a security parameter λ, index $w \in \{-k+2, -k+3, \ldots, n-1, n\}$ and outputs a decoder box D and two messages m_0, m_1.

Using the $\mathsf{MakeBox}$ experiment, we define the following probabilities, parameterized by $\gamma \in [0, 1/2]$ and $\mathsf{ind} \in \{0, 1, \ldots, k+1\}$, and a function of λ, w:

$$\Pr\text{-}\mathsf{Good\text{-}Dec}_{\mathcal{A},n,\gamma}(\lambda, w) = \Pr\left[D \text{ is } \gamma\text{-}\mathsf{Dist} \text{ for } m_0, m_1 \ : \ (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)\right]$$

$$\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\mathcal{A},n,\gamma}(\lambda, w) = \Pr\left[D \text{ is } \gamma\text{-}\mathsf{Dist}^{(\mathsf{ind})} \text{ for } m_0, m_1 \ : \ (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)\right]$$

We also define the following probability parameterized by $\gamma \in [0, 1/2]$ and $\mathsf{ind} \in \{1, \ldots, k\}$, and a function of λ, w:

$$\Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}_{\mathcal{A},n,\gamma}(\lambda, w) = \Pr\left[\exists\ \delta \in [0, 1/2] \text{ s.t. } \begin{array}{c} D \text{ is } \delta\text{-}\mathsf{Dist}^{(\mathsf{ind})} \wedge \\ D \text{ is not } (\delta - \gamma)\text{-}\mathsf{Dist}^{(\mathsf{ind}+1)} \end{array} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)\right]$$

Experiment $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)$

1. The challenger first samples $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{mBME.Setup}(1^\lambda, 1^{k+1})$, and sends pk to $\mathcal{A}$.
2. Let $S \subseteq [n]$ denote the set of indices for which $\mathcal{A}$ makes a key query. For each query $i \in S$, the challenger sends sk_i to $\mathcal{A}$ where sk_i is computed as follows:
   (a) if $i < w$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k+1})$,
   (b) if $w \le i < w + k$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k-i+w}1^{i-w+1})$,
   (c) if $i \ge w + k$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 1^{k+1})$.
3. The adversary outputs a decoder box D and messages m_0, m_1. The output of the experiment is (D, m_0, m_1).

Figure 10: Experiment $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)$

All the above probabilities are defined over all the random coins chosen by the challenger and adversary $\mathcal{A}$ during $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)$ experiment.

Next, we define the following events — $\mathsf{SubTr}^{(\mathsf{ind})}$ for $\mathsf{ind} \in [k]$. Recall that the Trace algorithm runs the $\mathsf{Subtrace}$ algorithm k times independently on indices 1 to k. Now whenever $\mathsf{Subtrace}$ algorithm outputs 1, then the Trace algorithm includes the corresponding index to the set of traitors. For $\mathsf{ind} \in [k]$, we say $\mathsf{SubTr}^{(\mathsf{ind})}$ occurs if the $\mathsf{Subtrace}$ algorithm outputs 1 when run for index ind. In other words, $\mathsf{SubTr}^{(\mathsf{ind})}$ occurs if ind^{th} run of $\mathsf{Subtrace}$ outputs 1.

We also define another event denoted as Tr. The event Tr occurs if the output of the Trace algorithm is a non-empty set. In other words, the event Tr is similar to $\mathsf{Cor\text{-}Tr}$, except that the output of the Trace algorithm is not required to be a subset of the set S of keys queried by $\mathcal{A}$. Now we know that $\mathsf{Tr} = \mathsf{Cor\text{-}Tr} \vee \mathsf{Fal\text{-}Tr}$, and $\mathsf{Tr} = \vee_{\mathsf{ind} \in [k]} \mathsf{SubTr}^{(\mathsf{ind})}$. The proof proceeds in the following steps. First, we show that $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}$ is related to $\Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}$ (for every $\mathsf{ind} \in \{1, \ldots, k\}$) via the following relation.

Note. Throughout this section, we will use $\widetilde{\epsilon}$ to denote the term $\epsilon/2(n+1)$.

Theorem 5.3. Let $\mathcal{A}$ be a PPT adversary, $n(\cdot), p(\cdot)$ polynomials, and $\epsilon(\cdot)$ a non-negligible function. If $\mathsf{mBME} = (\mathsf{mBME.Setup}, \mathsf{mBME.KeyGen}, \mathsf{mBME.Enc\text{-}PK}, \mathsf{mBME.Enc\text{-}SK}, \mathsf{mBME.Dec})$ satisfies *pk-sk ciphertext indistinguishability*, *key hiding* and *ciphertext hiding* properties (Definitions 4.1, 4.3 and 4.2), then there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, and every $\mathsf{ind} \in \{1, \ldots, k\}$,

$$\sum_{w} \Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}_{\mathcal{A},n,\widetilde{\epsilon}}(\lambda, w) \ge \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) - \mathrm{negl}(\lambda).$$

Let $\Pr\text{-}\mathsf{SubTr}^{(\mathsf{ind})}_{\mathcal{A},n,\epsilon}(\lambda)$ denote the probability of event $\mathsf{SubTr}^{(\mathsf{ind})}$ in the secure tracing experiment with adversary $\mathcal{A}$. Next, we show that $\Pr\text{-}\mathsf{SubTr}^{(\mathsf{ind})}_{\mathcal{A},n,\epsilon}$ is related to $\Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}$, and $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}$ is related to $\Pr\text{-}\mathsf{SubTr}^{(\mathsf{ind})}_{\mathcal{A},n,\epsilon}$ as follows.

Theorem 5.4. Let $\mathcal{A}$ be a PPT adversary, $n(\cdot), p(\cdot)$ polynomials and $\epsilon(\cdot)$ a non-negligible function. If $\mathsf{mBME} = (\mathsf{mBME.Setup}, \mathsf{mBME.KeyGen}, \mathsf{mBME.Enc\text{-}PK}, \mathsf{mBME.Enc\text{-}SK}, \mathsf{mBME.Dec})$ satisfies *ciphertext hiding* (Definition 4.2), then there exists a negligible functions $\mathrm{negl}_1(\cdot), \mathrm{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, and every $\mathsf{ind} \in \{1, \ldots, k\}$,

$$\Pr\text{-}\mathsf{SubTr}^{(\mathsf{ind})}_{\mathcal{A},n,\epsilon}(\lambda) \ge \frac{\left(\sum_w \Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}_{\mathcal{A},n,\widetilde{\epsilon}}(\lambda, w)\right)}{n(\lambda) + k - 1} - \mathrm{negl}_1(\lambda), \text{ and}$$

$$\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \ge 1 - \prod_{\mathsf{ind} \in [k]} \left(1 - \Pr\text{-}\mathsf{SubTr}^{(\mathsf{ind})}_{\mathcal{A},n,\epsilon}(\lambda)\right) - \mathrm{negl}_2(\lambda).$$

Observe that combining the above two theorems, we get the desired result that our scheme is a $\left(\frac{k}{n+k-1} - O\left(\frac{k(k-1)}{n^2}\right)\right)$-risky secure traitor tracing scheme. We want to point out that for $k = 1$ we get the tight *risky-ness*, i.e. $\frac{1}{n}$-risky secure scheme.

We will now prove these theorems in the following sections.

5.2.4 Proof of Theorem 5.3

For notational simplicity, we will skip the dependence of n and ϵ on λ. Also, we will skip the subscripts $\mathcal{A}$ and n when they are clear from the context.

Outline of the proof. Recall $\widetilde{\epsilon} = \epsilon/2(n+1)$. At a high level, this proof can be divided into the following steps:

- We first show that for every $\mathsf{ind} \in [k]$, $\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}}(2-\mathsf{ind}) \approx \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}$ and $\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(n+1)}(n+2-\mathsf{ind}) \approx 0$ (see Observation 5.1, Lemma 5.5 and Lemma 5.6).
- From this, it follows that for every $\mathsf{ind} \in [k]$, $\exists\ \Gamma_{\mathsf{ind}} \subseteq \{2-\mathsf{ind}, \ldots, n+1-\mathsf{ind}\}$ such that for all $w \in \Gamma_{\mathsf{ind}}$,
$$\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind}-1)}(w) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})}(w+1) > 0,$$
and the sum of these differences is at least $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon} - \text{negl}$ (see Observation 5.2).
- Next, we show that for every $\mathsf{ind} \in [k]$ and $w \in \{2-\mathsf{ind}, \ldots, n+1-\mathsf{ind}\}$, $\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind}+1)}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})+\widetilde{\epsilon}/2}(w) \approx \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})}(w+1)$ (see Lemma 5.7).
- After this, we relate $\Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}(w)$ to $\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}(w)$ and $\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind}+1)}(w)$. We show
$$\begin{aligned}\Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}_{\widetilde{\epsilon}/2}(w) &\geq \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind}-1)}(w) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind}+1)}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})+\widetilde{\epsilon}/2}(w) \text{ (see Lemma 5.8)}\\ &\approx \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind}-1)}(w) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})}(w+1)\end{aligned}$$
- As a result, we can conclude that for every $\mathsf{ind} \in [k]$,
$$\begin{aligned}\sum_w \Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}_{\widetilde{\epsilon}/2}(w) &\geq \sum_{w\in\Gamma_{\mathsf{ind}}} \Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}_{\widetilde{\epsilon}/2}(w)\\ &\geq \sum_{i\in\Gamma_{\mathsf{ind}}} \left(\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind}-1)}(w) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})}(w+1)\right)\\ &\geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon} - \text{negl}\end{aligned}$$

First, we have the following observation.

Observation 5.1. For every adversary $\mathcal{A}$, polynomial $n(\cdot)$ and $\lambda \in \mathbb{N}$, there exists an $w^* \in \{-k+2, \ldots, n(\lambda)\}$ such that $\Pr\text{-}\mathsf{Good\text{-}Dec}_{\mathcal{A},n,\epsilon}(\lambda, w^*) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda)$.

This observation simply follows from the fact that $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) = \frac{1}{n+k-1}\sum_w \Pr\text{-}\mathsf{Good\text{-}Dec}_{\mathcal{A},n,\epsilon}(\lambda, w)$, and therefore, there exists some index w^* such that $\Pr\text{-}\mathsf{Good\text{-}Dec}_{\mathcal{A},n,\epsilon}(\lambda, w^*) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda)$.

Lemma 5.5. If $\mathsf{mBME} = (\mathsf{mBME.Setup}, \mathsf{mBME.KeyGen}, \mathsf{mBME.Enc\text{-}PK}, \mathsf{mBME.Enc\text{-}SK}, \mathsf{mBME.Dec})$ satisfies *pk-sk ciphertext indistinguishability*, *key hiding* and *ciphertext hiding*, then for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, every $\mathsf{ind} \in [k]$,
$$\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}}(\lambda, 2-\mathsf{ind}) \geq \Pr\text{-}\mathsf{Good\text{-}Dec}_{\epsilon}(\lambda, w^*) - \text{negl}(\lambda) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) - \text{negl}(\lambda).$$

Proof. The proof of above theorem is independent of the choice of index ind. Thus, we treat ind as a fixed value throughout the proof.

Let $w = 2 - \mathsf{ind}$, and $\gamma_j = \epsilon - j \cdot \widetilde{\epsilon}/(w^* + w + 2k)$ for $0 \leq j \leq w^* + w + 2k$. Also, let $\rho_0(\lambda) = \Pr\text{-}\mathsf{Good\text{-}Dec}_{\epsilon}(\lambda, w^*)$ and $\rho_{w^*+w+2k}(\lambda) = \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}}(\lambda, w)$. In order to show that $\rho_0(\lambda) - \rho_{w^*+w+2k}(\lambda)$ is negligible in λ, we will define a sequence of hybrid experiments, and show that the difference in the consecutive probabilities is at most negligible in λ.

Hybrid Hyb_0. This experiment corresponds to the $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w^*)$, where $\mathcal{A}$ finally outputs a decoder box D and two messages m_0, m_1 such that D can distinguish between standard encryptions of m_0 and m_1 with advantage at least γ_0. Now note that $\rho_0(\lambda)$ denotes the probability that $\mathcal{A}$ outputs a decoder box D that is γ_0-Dist for m_0, m_1.

Hybrid Hyb_j for $1 \leq j \leq w^* + k - 1$. This experiment is identical to Hyb_0, except the challenger now answers the secret key queries for first j users corresponding to vector 1^{k+1}. Concretely, for each key query i, the challenger sends sk_i to $\mathcal{A}$ where sk_i is computed as follows:

1. if $i \leq j$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 1^{k+1})$,
2. if $j < i < w^*$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k+1})$,
3. if $\max(j+1, w^*) \leq i < w^* + k$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k-i+w^*} 1^{i-w^*+1})$,
4. if $i \geq w^* + k$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 1^{k+1})$.

Let $\rho_j(\lambda)$ denote the probability that $\mathcal{A}$ outputs a decoder box D that is γ_j-Dist for m_0, m_1.

Hybrid Hyb_j for $w^* + k \leq j < w^* + w + 2k - 1$. This experiment is identical to Hyb_{w^*+k-1}, except the challenger now answers the secret key queries for first $(j - w^* - k + 1)$ users as it would in $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)$. Concretely, for each key query i, the challenger sends sk_i to $\mathcal{A}$ where sk_i is computed as follows:

1. if $i < \min(w, j - w^* - k + 2)$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k+1})$,
2. if $w \leq i \leq j - w^* - k + 2$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k-i+w} 1^{i-w+1})$,
3. if $i > j - w^* - k + 2$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 1^{k+1})$.

Let $\rho_j(\lambda)$ denote the probability that $\mathcal{A}$ outputs a decoder box D that is γ_j-Dist for m_0, m_1.

Hybrid $\mathsf{Hyb}_{w^*+w+2k-1}$. This experiment is identical to $\mathsf{Hyb}_{w^*+w+2k-2}$, except now $\rho_{w^*+w+2k-1}(\lambda)$ denotes the probability that $\mathcal{A}$ outputs a decoder box D that is $\gamma_{w^*+w+2k-1}$-$\mathsf{Dist}^{(0)}$ for m_0, m_1.

Hybrid Hyb_{w^*+w+2k}. This experiment is identical to $\mathsf{Hyb}_{w^*+w+2k-1}$, except now $\rho_{w^*+w+2k}(\lambda)$ denotes the probability that $\mathcal{A}$ outputs a decoder box D that is γ_{w^*+w+2k}-$\mathsf{Dist}^{(\mathsf{ind})}$ for m_0, m_1. In other words, this corresponds to the $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)$ experiment.

Claim 5.1. If mBME satisfies *key hiding*, then for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $0 \leq j < w^* + w + 2k - 2$, $\rho_j(\lambda) - \rho_{j+1}(\lambda) \leq \text{negl}(\lambda)$.

Proof. Fix any value j. Now it is sufficient to show that $\rho_j(\lambda) - \rho_{j+1}(\lambda)$ is bounded by a negligible function. Suppose, on the contrary, there exists a PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible functions $\epsilon(\cdot), \delta(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/q(\lambda)$, $\rho_j(\lambda) - \rho_{j+1}(\lambda) \geq \delta(\lambda)$. Then we can construct an algorithm $\mathcal{B}$ that can break key hiding property of the underlying mixed bit matching encryption scheme.

The reduction algorithm $\mathcal{B}$ sends the challenge attribute pair $(\mathbf{x}_0^*, \mathbf{x}_1^*)$ to the mixed bit matching encryption scheme challenger where $\mathbf{x}_0^*$ and $\mathbf{x}_1^*$ are computed as follows:

$$\mathbf{x}_0^* = \begin{cases} 0^{k+1} & \text{if } j < w^* - 1, \\ 0^{k-j+w^*} 1^{j-w^*+1} & \text{if } w^* - 1 \leq j < w^* + k - 1, , \\ 1^{k+1} & \text{otherwise.} \end{cases}$$

$$\mathbf{x}_1^* = \begin{cases} 1^{k+1} & \text{if } j < w^* + k - 1, \\ 0^{k+1} & \text{if } w^* + k - 1 \leq j < w^* + w + k - 2, \\ 0^{k-j+(w^*+w+k-2)} 1^{j-(w^*+w+k-2)+1} & \text{otherwise.} \end{cases}$$

The mBME challenger samples mBME key pair $(\mathsf{pk}, \mathsf{msk})$, and sends pk to the reduction algorithm $\mathcal{B}$ which it forwards to adversary $\mathcal{A}$. The challenger also sends the challenge key sk^* to $\mathcal{B}$. Next, $\mathcal{A}$ makes key queries to $\mathcal{B}$ which are answered as follows.

Case 1: $j < w^* + k - 1$. For each key query $i \in [n]$, (1) if $i \leq j$, then $\mathcal{B}$ makes secret key query for vector 1^{k+1} to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$; (2) else if $i = j+1$, then $\mathcal{B}$ sends sk^* to $\mathcal{A}$; (3) else if $j+1 < i < w^*$, then $\mathcal{B}$ makes secret key query for vector 0^{k+1} to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$; (4) else if $\max(j+2, w^*) \leq i < w^* + k$, then $\mathcal{B}$ makes secret key query for vector $0^{k-i+w^*} 1^{i-w^*+1}$ to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$; (5) otherwise $\mathcal{B}$ makes secret key query for vector 1^{k+1} to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$.

Case 2: $j \geq w^* + k - 1$. For each key query $i \in [n]$, (1) if $i < \min(w, j - w^* - k + 2)$, then $\mathcal{B}$ makes secret key query for vector 0^{k+1} to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$; (2) else if $w \leq i < j - w^* - k + 2$, then $\mathcal{B}$ makes secret key query for vector $0^{k-i+w} 1^{i-w+1}$ to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$; (3) else if $i = j - w^* - k + 2$, then $\mathcal{B}$ sends sk^* to $\mathcal{A}$; (4) otherwise $\mathcal{B}$ makes secret key query for vector 1^{k+1} to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$.

After all these queries, the adversary $\mathcal{A}$ sends a decoder box D and messages m_0, m_1 to $\mathcal{B}$. The reduction algorithm $\mathcal{B}$ sets $\gamma = (\gamma_j + \gamma_{j+1})/2$, $T = \lambda \cdot n/\epsilon$ and tests whether D is a γ-Dist box for m_0, m_1 using simple counting based estimation. Concretely, it first sets $\mathsf{count} = 0$. For $\ell = 1$ to T, it chooses $b_\ell \leftarrow \{0,1\}$, computes $\mathsf{ct}_\ell \leftarrow \mathsf{mBME.Enc\text{-}PK}(\mathsf{pk}, m_{b_\ell})$. Next, if $D(\mathsf{ct}_\ell) = b_\ell$, it sets $\mathsf{count} = \mathsf{count} + 1$, else it sets $\mathsf{count} = \mathsf{count} - 1$. Finally, after the T iterations, if $\mathsf{count} > \gamma \cdot T$, then $\mathcal{B}$ guesses 0 (i.e., sk^* corresponds to $\mathbf{x}_0^*$), else it guesses 1 (i.e., sk^* corresponds to $\mathbf{x}_1^*$).

First, note that $\mathcal{B}$ is an admissible adversary in the key hiding game since it does not make any secret-key encryption queries. Now we analyse $\mathcal{B}$'s advantage in breaking key hiding security. Let b and b' denote the challenger's bit and $\mathcal{B}$'s guess, respectively. First, we show that $\Pr[b' = 0 \mid b = 0] \geq \rho_j(\lambda) - \text{negl}(\lambda)$.

$$\begin{aligned} \Pr[b' = 0 \mid b = 0] &= \Pr[\mathsf{count} > \gamma \cdot T \mid b = 0] \\ &\geq \Pr[\mathcal{A} \text{ outputs } \gamma_j\text{-}\mathsf{Dist} \text{ box } D \wedge \mathsf{count} > \gamma \cdot T \mid b = 0] \\ &\geq \Pr[\mathcal{A} \text{ outputs } \gamma_j\text{-}\mathsf{Dist} \text{ box } D \mid b = 0] \\ &\quad - \Pr[\mathcal{A} \text{ outputs } \gamma_j\text{-}\mathsf{Dist} \text{ box } D \wedge \mathsf{count} \leq \gamma \cdot T \mid b = 0] \\ &\geq \rho_j(\lambda) - \Pr[\mathsf{count} \leq \gamma \cdot T \mid b = 0 \wedge \mathcal{A} \text{ outputs } \gamma_j\text{-}\mathsf{Dist} \text{ box } D] \\ &\geq \rho_j(\lambda) - 2^{-O(\lambda)}. \end{aligned}$$

The last inequality follows by applying a Chernoff bound similar to that used in Lemma 5.3. Next, we show that $\Pr[b' = 0 \mid b = 1] \leq \rho_{j+1}(\lambda) + \mathrm{negl}(\lambda)$.

$$\begin{aligned}\Pr[b' = 0 \mid b = 1] &= \Pr[\mathsf{count} > \gamma \cdot T \mid b = 1] \\ &\leq \Pr[\mathcal{A} \text{ does not output } \gamma_{j+1}\text{-}\mathsf{Dist} \text{ box } D \wedge \mathsf{count} > \gamma \cdot T \mid b = 1] \\ &\quad + \Pr[\mathcal{A} \text{ outputs } \gamma_{j+1}\text{-}\mathsf{Dist} \text{ box } D \wedge \mathsf{count} > \gamma \cdot T \mid b = 1] \\ &\leq \Pr[\mathsf{count} > \gamma \cdot T \mid b = 1 \wedge \mathcal{A} \text{ does not output } \gamma_{j+1}\text{-}\mathsf{Dist} \text{ box } D] + \rho_{j+1}(\lambda) \\ &\leq 2^{-O(\lambda)} + \rho_{j+1}(\lambda).\end{aligned}$$

As before, the last inequality follows by applying a Chernoff bound. Thus, combining above bounds, we get that

$$\begin{aligned}\Pr[\mathcal{B} \text{ wins}] &= \Pr[b' = 0 \mid b = 0] - \Pr[b' = 0 \mid b = 1] \\ &\geq \rho_j(\lambda) - \rho_{j+1}(\lambda) - \mathrm{negl}(\lambda) \geq \delta(\lambda) - \mathrm{negl}(\lambda).\end{aligned}$$

Since mBME satisfies key hiding, thus we get a contradiction. Hence, the claim follows. ■

Claim 5.2. If mBME satisfies *pk-sk ciphertext indistinguishability*, then for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_{w^*+w+2k-2}(\lambda) - \rho_{w^*+w+2k-1}(\lambda) \leq \mathrm{negl}(\lambda)$.

Proof. Let $j = w^* + w + 2k - 2$. Suppose, on the contrary, there exists a PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible functions $\epsilon(\cdot), \delta(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/q(\lambda)$, $\rho_j(\lambda) - \rho_{j+1}(\lambda) \geq \delta(\lambda)$. Then we can construct an algorithm $\mathcal{B}$ that can break pk-sk ciphertext indistinguishability property of the underlying mixed bit matching encryption scheme.

The mBME challenger samples mBME key pair $(\mathsf{pk}, \mathsf{msk})$, and sends pk to the reduction algorithm $\mathcal{B}$ which it forwards to adversary $\mathcal{A}$. Next, $\mathcal{A}$ makes key queries to $\mathcal{B}$ which are answered as in $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)$. That is, for each key query $i \in [n]$, $\mathcal{B}$ makes secret key query for vector $\mathbf{x}_i$ to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$. (Here $\mathbf{x}_i$ is as defined in the construction. Note that $\mathbf{x}_i$ depends on the index w.) After all these queries, the adversary $\mathcal{A}$ sends a decoder box D and messages m_0, m_1 to $\mathcal{B}$.

$\mathcal{B}$ then chooses two bits α, β uniformly at random, i.e. $\alpha, \beta \leftarrow \{0, 1\}$. Next, $\mathcal{B}$ sends message m_α as its challenge message. It receives challenge ciphertext ct^* from mBME challenger. $\mathcal{B}$ then also queries the mBME challenger for a (secret-key) encryption of m_α for vector 1^{k+1}. Let ct_1 be the challenger's response. It also computes ciphertext ct_0 as $\mathsf{ct}_0 \leftarrow \mathsf{mBME.Enc\text{-}PK}(\mathsf{pk}, m_\alpha)$. Finally, $\mathcal{B}$ runs decoder box D on ct_β and ct^* independently, and if $D(\mathsf{ct}_\beta) = D(\mathsf{ct}^*)$, it outputs $b' = \beta$, else it outputs $b' = 1 - \beta$ as its guess.

Now we analyse $\mathcal{B}$'s advantage in breaking pk-sk ciphertext indistinguishability security. First, note that $\mathcal{B}$ perfectly simulates hybrids j and $j + 1$ for adversary $\mathcal{A}$, and in both these hybrids, the challenger and adversary are interacting as in $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)$ experiment. Therefore, we can write that

$$\begin{aligned}\rho_j(\lambda) &= \Pr\text{-}\mathsf{Good\text{-}Dec}_{\mathcal{A},n,\gamma_j}(\lambda, w) \\ &= \Pr\left[D \text{ is } \gamma_j\text{-}\mathsf{Dist} \text{ for } m_0, m_1 \ : \ (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)\right] \\ \rho_{j+1}(\lambda) &= \Pr\text{-}\mathsf{Good\text{-}Dec}^{(0)}_{\mathcal{A},n,\gamma_{j+1}}(\lambda, w) \\ &= \Pr\left[D \text{ is } \gamma_{j+1}\text{-}\mathsf{Dist}^{(0)} \text{ for } m_0, m_1 \ : \ (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)\right]\end{aligned}$$

Let $p_D = \Pr[D(\mathsf{ct}) = b : \mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}PK}(\mathsf{pk}, m_b)]$, and $p_D^{(0)} = \Pr[D(\mathsf{ct}) = b : \mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}SK}(\mathsf{msk}, m_b, 1^{k+1})]$, where the probability is taken over bit b, random coins of decoder D and randomness used during encryption. Let $\mathsf{Diff\text{-}Adv}$ denote the event when the advantage of decoder D in distinguishing between public-key encryptions of m_0 and m_1 is $\widetilde{\epsilon}/(w^* + w + 2k)$ more than its advantage in distinguishing secret-key encryptions (w.r.t vector 1^{k+1}). Formally, $\mathsf{Diff\text{-}Adv} \ : \ p_D - p_D^{(0)} > \widetilde{\epsilon}/(w^* + w + 2k)$. Recall that we have $\rho_j(\lambda) - \rho_{j+1}(\lambda) \geq \delta(\lambda)$.

Thus, we can write the following:

$$\begin{aligned}\Pr[\mathsf{Diff\text{-}Adv}] &\geq \Pr\left[\begin{array}{c} D \text{ is } \gamma_j\text{-}\mathsf{Dist} \wedge \\ D \text{ is not } \gamma_{j+1}\text{-}\mathsf{Dist}^{(0)} \end{array} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)\right] \\ &\geq \Pr\text{-}\mathsf{Good\text{-}Dec}_{\mathcal{A},n,\gamma_j}(\lambda, w) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{(0)}_{\mathcal{A},n,\gamma_{j+1}}(\lambda, w) \\ &\geq \rho_j(\lambda) - \rho_{j+1}(\lambda) \geq \delta(\lambda)\end{aligned}$$

The remaining analysis is similar to that provided for proving Lemma 5.4. Finally, we get that $\Pr[\mathcal{B} \text{ wins}] \geq \frac{1}{2} + \frac{\delta}{2} \cdot \left(\frac{\widetilde{\epsilon}}{w^* + w + 2k}\right)^2$. Thus, the claim follows. ■

Claim 5.3. If mBME satisfies *ciphertext hiding*, then for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_{w^*+w+2k-1}(\lambda) - \rho_{w^*+w+2k}(\lambda) \leq \text{negl}(\lambda)$.

Proof. The proof of this lemma is similar to that of Claim 5.2. We briefly highlight the changes. First, the reduction algorithm $\mathcal{B}$ sends challenge attribute pair as $(\mathbf{y}_0, \mathbf{y}_1) = (1^{k+1}, 1^{k+1-\mathsf{ind}}0^{\mathsf{ind}})$ to the mixed bit matching encryption scheme challenger at the start. Second, $\mathcal{B}$ queries the mBME challenger for a (secret-key) encryption of m_α for vector $\mathbf{y}_\beta$ (where $\alpha, \beta \leftarrow \{0,1\}$), and sets the response as ciphertext ct. Note that in previous proof, $\mathcal{B}$ generates two ciphertexts ct_0 (by running $\mathsf{mBME.Enc\text{-}PK}$) and ct_1 (by querying challenger), and later chooses one of them at random to perform the final check ($D(\mathsf{ct}_\beta) = D(\mathsf{ct}^*)$). Here it will perform the check as $D(\mathsf{ct}) = D(\mathsf{ct}^*)$. Finally, since for all keys sk_i (corresponding to vector $\mathbf{x}_i$) given to the adversary, we have that $f(\mathbf{x}_i, \mathbf{y}_0) = f(\mathbf{x}_i, \mathbf{y}_1) = 1$ (where f denotes the 'AND-of-ORs'). This can be verified by just checking that $f(\mathbf{x}_1, \mathbf{y}_0) = f(\mathbf{x}_1, \mathbf{y}_1) = 1$ holds. Since $\mathbf{x}_1 = 0^{k+1-\mathsf{ind}}1^{\mathsf{ind}}$, thus the invariant holds which itself implies that $f(\mathbf{x}_i, \mathbf{y}_0) = f(\mathbf{x}_i, \mathbf{y}_1) = 1$ for $i \geq 1$. Thus, the reduction algorithm $\mathcal{B}$ will be an admissible adversary in the ciphertext hiding game. Remaining proof is identical therefore we skip it. ■

■

Lemma 5.6. If $\mathsf{mBME} = (\mathsf{mBME.Setup}, \mathsf{mBME.KeyGen}, \mathsf{mBME.Enc\text{-}PK}, \mathsf{mBME.Enc\text{-}SK}, \mathsf{mBME.Dec})$ satisfies *ciphertext hiding*, then for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, every $\mathsf{ind} \in [k]$,

$$\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\mathcal{A},n,\epsilon-\widetilde{\epsilon}\cdot(n+1)}(\lambda, n+2-\mathsf{ind}) \leq \text{negl}(\lambda).$$

Proof. Fix any index ind. Suppose, on the contrary, there exists a PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible functions $\epsilon(\cdot), \delta(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/q(\lambda)$, $\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\mathcal{A},n,\epsilon-\widetilde{\epsilon}\cdot(n+1)}(\lambda, n+2-\mathsf{ind}) \geq \delta(\lambda)$. Then we can construct an algorithm $\mathcal{B}$ that can break ciphertext hiding property of the underlying mixed bit matching encryption scheme.

The reduction algorithm $\mathcal{B}$ sends challenge attribute pair as $\mathbf{y}_0 = \mathbf{y}_1 = 1^{k+1-\mathsf{ind}}0^{\mathsf{ind}}$ to the mixed bit matching encryption scheme challenger. The mBME challenger samples mBME key pair $(\mathsf{pk}, \mathsf{msk})$, and sends pk to the reduction algorithm $\mathcal{B}$ which it forwards to adversary $\mathcal{A}$. Next, $\mathcal{A}$ makes key queries to $\mathcal{B}$ which are answered as in $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, n+2-\mathsf{ind})$. That is, for each key query $i \in [n]$, $\mathcal{B}$ makes secret key query for vector $\mathbf{x}_i$ to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$. (Here $\mathbf{x}_i$ is as defined in the construction. Note that $\mathbf{x}_i$ depends on the index $w = n + 2 - \mathsf{ind}$.) After all these queries, the adversary $\mathcal{A}$ sends a decoder box D and messages m_0, m_1 to $\mathcal{B}$.

$\mathcal{B}$ then sends message (m_0, m_1) as its challenge messages. It receives challenge ciphertext ct^* from mBME challenger. $\mathcal{B}$ then chooses a random bit $\beta \leftarrow \{0,1\}$, and queries the mBME challenger for a (secret-key) encryption of m_β for vector $\mathbf{y}_0$. Let ct be the challenger's response. Finally, $\mathcal{B}$ runs decoder box D on ct and ct^* independently, and if $D(\mathsf{ct}) = D(\mathsf{ct}^*)$, it outputs $b' = \beta$, else it outputs $b' = 1 - \beta$ as its guess.

The probability analysis of $\mathcal{B}$'s advantage is similar to that provided for Claim 5.2, however we need to argue that $\mathcal{B}$ is an admissible adversary. For admissibility, we need to show that none of the keys sk_i queried by $\mathcal{B}$ can decrypt the challenge ct^*. That is, for all i, $f(\mathbf{x}_i, \mathbf{y}_0) = 0$. Now it is sufficient to show that $f(\mathbf{x}_n, \mathbf{y}_0) = 0$. Note that $\mathbf{y}_0 = 1^{k+1-\mathsf{ind}}0^{\mathsf{ind}}$, and $\mathbf{x}_n = 0^{k+2-\mathsf{ind}}1^{\mathsf{ind}-1}$. Therefore, we get that $f(\mathbf{x}_n, \mathbf{y}_0) = 0$. Thus, $\mathcal{B}$ is an admissible adversary and the lemma follows. ■

From the above lemmas, it follows that $\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}}(\lambda, 2-\mathsf{ind}) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(n+1)}(\lambda, n+2-\mathsf{ind}) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) - \mathrm{negl}(\lambda)$. This brings us to the following observation.

Observation 5.2. If mBME satisfies *pk-sk ciphertext indistinguishability*, *key hiding* and *ciphertext hiding*, then for any PPT adversary $\mathcal{A}$, non-negligible function $\epsilon(\cdot)$, polynomials $n(\cdot), p(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, and every $\mathsf{ind} \in [k]$, there exists a subset $\Gamma_{\mathsf{ind}} \subseteq \{2-\mathsf{ind}, \ldots, n+1-\mathsf{ind}\}$ such that for every $w \in \Gamma_{\mathsf{ind}}$, $\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind}-1)}(\lambda, w) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})}(\lambda, w+1) > 0$ and

$$\sum_{w \in \Gamma_{\mathsf{ind}}} \left(\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind}-1)}(\lambda, w) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})}(\lambda, w+1) \right) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) - \mathrm{negl}(\lambda).$$

The next lemma will prove that $\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}(w+1)$ and $\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind}+1)}(w)$ are approximately equal for all w, ind.

Lemma 5.7. If $\mathsf{mBME} = (\mathsf{mBME.Setup}, \mathsf{mBME.KeyGen}, \mathsf{mBME.Enc\text{-}PK}, \mathsf{mBME.Enc\text{-}SK}, \mathsf{mBME.Dec})$ satisfies *key hiding* and *ciphertext hiding* properties, then for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, every $\mathsf{ind} \in [k]$ and $w \in \{2-\mathsf{ind}, \ldots, n+1-\mathsf{ind}\}$,

$$\Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind}+1)}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})+\widetilde{\epsilon}/2}(\lambda, w) \leq \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})}(\lambda, w+1) + \mathrm{negl}(\lambda).$$

Proof. The proof of above theorem is independent of the choice of index ind and w. Thus, we treat ind and w as a fixed value throughout the proof.

Let $\gamma_j = \epsilon - \widetilde{\epsilon}\cdot(w+\mathsf{ind}) + \widetilde{\epsilon}/2 - j\cdot\widetilde{\epsilon}/(2k+4)$ for $0 \leq j \leq k+2$. Also, let $\rho_0(\lambda) = \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind}+1)}_{\gamma_0}(\lambda, w)$ and $\rho_{k+2}(\lambda) = \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\gamma_{k+2}}(\lambda, w+1)$. In order to show that $\rho_0(\lambda) - \rho_{k+2}(\lambda)$ is negligible in λ, we will define a sequence of hybrid experiments, and show that the difference in the consecutive probabilities is at most negligible in λ.

Hybrid Hyb_0. This experiment corresponds to the $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)$, where $\mathcal{A}$ finally outputs a decoder box D and two messages m_0, m_1 such that D can distinguish between index-$(\mathsf{ind}+1)$ encryptions of m_0 and m_1 with advantage at least γ_0. Now note that $\rho_0(\lambda)$ denotes the probability that $\mathcal{A}$ outputs a decoder box D that is γ_0-$\mathsf{Dist}^{(\mathsf{ind}+1)}$ for m_0, m_1.

Hybrid Hyb_j for $1 \leq j \leq \mathsf{ind}$. This experiment is identical to Hyb_0, except the key queries for indices $< w+j$ are answered as per $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w+1)$. Concretely, for each key query i, the challenger sends sk_i to $\mathcal{A}$ where sk_i is computed as follows:

1. if $i < w$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k+1})$,
2. if $w \leq i < w+j$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k-i+w+1}1^{i-w})$,
3. if $w+j \leq i < w+k+1$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k-i+w}1^{i-w+1})$,
4. if $i \geq w+k+1$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 1^{k+1})$.

Let $\rho_j(\lambda)$ denote the probability that $\mathcal{A}$ outputs a decoder box D that is γ_j-$\mathsf{Dist}^{(\mathsf{ind}+1)}$ for m_0, m_1.

Hybrid $\mathsf{Hyb}_{\mathsf{ind}+1}$. This experiment is identical to $\mathsf{Hyb}_{\mathsf{ind}}$, except now $\rho_{\mathsf{ind}+1}(\lambda)$ denotes the probability that $\mathcal{A}$ outputs a decoder box D that is $\gamma_{\mathsf{ind}+1}\text{-}\mathsf{Dist}^{(\mathsf{ind})}$ for m_0, m_1.

Hybrid Hyb_{j+1} for $\mathsf{ind}+1 \leq j \leq k+1$. This experiment is identical to $\mathsf{Hyb}_{\mathsf{ind}+1}$, except the key queries for indices $< w+j$ are answered as per $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w+1)$. Concretely, for each key query i, the challenger sends sk_i to $\mathcal{A}$ where sk_i is computed as follows:

1. if $i < w+1$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k+1})$,
2. if $w+1 \leq i < w+j$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k-i+w+1}1^{i-w})$,
3. if $w+j \leq i < w+k+1$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 0^{k-i+w}1^{i-w+1})$,
4. if $i \geq w+k+1$, then $\mathsf{sk}_i \leftarrow \mathsf{mBME.KeyGen}(\mathsf{msk}, 1^{k+1})$.

Let $\rho_{j+1}(\lambda)$ denote the probability that $\mathcal{A}$ outputs a decoder box D that is $\gamma_{j+1}\text{-}\mathsf{Dist}^{(\mathsf{ind})}$ for m_0, m_1. Note that Hyb_{k+2} corresponds to the $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w+1)$ experiment.

Claim 5.4. If mBME satisfies *key hiding*, then for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $0 \leq j < \mathsf{ind}$, $\rho_j(\lambda) - \rho_{j+1}(\lambda) \leq \mathrm{negl}(\lambda)$.

Proof. Fix any value j. Now it is sufficient to show that $\rho_j(\lambda) - \rho_{j+1}(\lambda)$ is bounded by a negligible function. Suppose, on the contrary, there exists a PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible functions $\epsilon(\cdot), \delta(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/q(\lambda)$, $\rho_j(\lambda) - \rho_{j+1}(\lambda) \geq \delta(\lambda)$. Then we can construct an algorithm $\mathcal{B}$ that can break key hiding property of the underlying mixed bit matching encryption scheme.

The reduction algorithm $\mathcal{B}$ sends the challenge attribute pair $(\mathbf{x}_0^*, \mathbf{x}_1^*) = (0^{k-j}1^{j+1}, 0^{k-j+1}1^{j})$ to the mixed bit matching encryption scheme challenger. The mBME challenger samples mBME key pair $(\mathsf{pk}, \mathsf{msk})$, and sends pk to the reduction algorithm $\mathcal{B}$ which it forwards to adversary $\mathcal{A}$. The challenger also sends the challenge key sk^* to $\mathcal{B}$. Next, $\mathcal{A}$ makes key queries to $\mathcal{B}$ which are answered as follows.

For each key query $i \in [n]$, (1) if $i < w$, then $\mathcal{B}$ makes secret key query for vector 0^{k+1} to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$; (2) else if $w \leq i < w+j$, then $\mathcal{B}$ makes secret key query for vector $0^{k-i+w+1}1^{i-w}$ to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$; (3) else if $i = w+j$, then $\mathcal{B}$ sends sk^* to $\mathcal{A}$; (4) else if $w+j < i < w+k+1$, then $\mathcal{B}$ makes secret key query for vector $0^{k-i+w}1^{i-w+1}$ to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$; (5) otherwise $\mathcal{B}$ makes secret key query for vector 1^{k+1} to mBME challenger and forwards the challenger's response sk_i to $\mathcal{A}$.

After all these queries, the adversary $\mathcal{A}$ sends a decoder box D and messages m_0, m_1 to $\mathcal{B}$. The reduction algorithm $\mathcal{B}$ sets $\gamma = (\gamma_j + \gamma_{j+1})/2$, $T = \lambda \cdot n/\epsilon$ and tests whether D is a γ-Dist box for m_0, m_1 using simple counting based estimation. Concretely, it first sets $\mathsf{count} = 0$. For $\ell = 1$ to T, it chooses $b_\ell \leftarrow \{0,1\}$ and sends $(m_{b_\ell}, 1^{k-\mathsf{ind}}0^{\mathsf{ind}+1})$ for secret key encryption to the mBME challenger, and receives ciphertext ct_ℓ as response. Next, if $D(\mathsf{ct}_\ell) = b_\ell$, it sets $\mathsf{count} = \mathsf{count} + 1$, else it sets $\mathsf{count} = \mathsf{count} - 1$. Finally, after the T iterations, if $\mathsf{count} > \gamma \cdot T$, then $\mathcal{B}$ guesses 0 (i.e., sk^* corresponds to $\mathbf{x}_0$), else it guesses 1 (i.e., sk^* corresponds to $\mathbf{x}_1$).

The probability analysis of $\mathcal{B}$'s advantage is similar to that provided for Claim 5.1, however we need to argue that $\mathcal{B}$ is an admissible adversary. For admissibility, we need to show that the decryptability of the secret-key ciphertexts ct_ℓ by the challenge secret key sk^* is independent of whether challenger used $\mathbf{x}_0^*$ or $\mathbf{x}_1^*$. That is, for all i, $f(\mathbf{x}_0^*, 1^{k-\mathsf{ind}}0^{\mathsf{ind}+1}) = f(\mathbf{x}_1^*, 1^{k-\mathsf{ind}}0^{\mathsf{ind}+1})$. Note that $\mathbf{x}_0^* = 0^{k-j}1^{j+1}$, and $\mathbf{x}_1^* = 0^{k-j+1}1^{j}$. Since $0 \leq j < \mathsf{ind}$, we get that $f(\mathbf{x}_b^*, 1^{k-\mathsf{ind}}0^{\mathsf{ind}+1}) = 0$ for both $b \in \{0,1\}$. Thus, $\mathcal{B}$ is an admissible adversary and the claim follows. ■

Claim 5.5. If mBME satisfies *ciphertext hiding*, then for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_{\mathsf{ind}}(\lambda) - \rho_{\mathsf{ind}+1}(\lambda) \leq \mathrm{negl}(\lambda)$.

Proof. The proof of this lemma is similar to the proof of Claim 5.3. We briefly highlight the changes. First, the reduction algorithm $\mathcal{B}$ sends challenge attribute pair as $(\mathbf{y}_0, \mathbf{y}_1) = (1^{k+1-\mathsf{ind}}0^{\mathsf{ind}}, 1^{k-\mathsf{ind}}0^{\mathsf{ind}+1})$ to the mixed bit matching encryption scheme challenger at the start. Second, $\mathcal{B}$ queries the mBME challenger for a (secret-key) encryption of m_α for vector $\mathbf{y}_\beta$ (where $\alpha, \beta \leftarrow \{0,1\}$), and sets the response as ciphertext ct. Finally, since for all keys sk_i (corresponding to vector $\mathbf{x}_i$) given to the adversary, we have that $f(\mathbf{x}_i, \mathbf{y}_0) = f(\mathbf{x}_i, \mathbf{y}_1)$ (where f denotes the 'AND-of-ORs'). We claim that $f(\mathbf{x}_i, \mathbf{y}_0) = f(\mathbf{x}_i, \mathbf{y}_1) = 0$ for $i < w + \mathsf{ind}$, and $= 1$ for $i \geq w + \mathsf{ind}$. This can be verified by just checking that $f(\mathbf{x}_{w+\mathsf{ind}-1}, \mathbf{y}_b) = 0$ and $f(\mathbf{x}_{w+\mathsf{ind}}, \mathbf{y}_0) = 1$ for both $b \in \{0,1\}$. Since $\mathbf{x}_{w+\mathsf{ind}-1} = 0^{k+2-\mathsf{ind}}1^{\mathsf{ind}-1}$ and $\mathbf{x}_{w+\mathsf{ind}} = 0^{k-\mathsf{ind}}1^{\mathsf{ind}+1}$, thus we satisfy the required constraint. Thus, the reduction algorithm $\mathcal{B}$ will be an admissible adversary in the ciphertext hiding game. Remaining proof is identical therefore we skip it. ∎

Claim 5.6. If mBME satisfies *key hiding*, then for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\mathsf{ind} + 1 \leq j < k + 2$, $\rho_j(\lambda) - \rho_{j+1}(\lambda) \leq \mathrm{negl}(\lambda)$.

Proof. The proof of this lemma is similar to the proof of Claim 5.4. ∎

∎

Lemma 5.8. For any PPT adversary $\mathcal{A}$, polynomial $n(\cdot)$ and non-negligible function $\epsilon(\cdot)$, all $\lambda \in \mathbb{N}$, every $\mathsf{ind} \in [k]$ and $w \in \{2 - \mathsf{ind}, \ldots, n + 1 - \mathsf{ind}\}$,

$$\Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}_{\widetilde{\epsilon}/2}(\lambda, w) \geq \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind}-1)}(\lambda, w) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind}+1)}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})+\widetilde{\epsilon}/2}(\lambda, w).$$

Proof. Fix any index ind and w. Recall that $\Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}$ is defined as below

$$\Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}_{\mathcal{A},n,\widetilde{\epsilon}/2}(\lambda, w) = \Pr\left[\exists\, \delta \in [0, 1/2] \text{ s.t. } \begin{array}{c} D \text{ is } \delta\text{-}\mathsf{Dist}^{(\mathsf{ind})} \wedge \\ D \text{ is not } (\delta - \widetilde{\epsilon}/2)\text{-}\mathsf{Dist}^{(\mathsf{ind}+1)} \end{array} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)\right].$$

Now we can also write that

$$\Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}_{\mathcal{A},n,\widetilde{\epsilon}/2}(\lambda, w) \geq \max_{\delta \in [0,1/2]} \Pr\left[\begin{array}{c} D \text{ is } \delta\text{-}\mathsf{Dist}^{(\mathsf{ind})} \wedge \\ D \text{ is not } (\delta - \widetilde{\epsilon}/2)\text{-}\mathsf{Dist}^{(\mathsf{ind}+1)} \end{array} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)\right].$$

We also know that for any $\delta \in [0, 1/2]$,

$$\begin{aligned} &\Pr\left[\begin{array}{c} D \text{ is } \delta\text{-}\mathsf{Dist}^{(\mathsf{ind})} \wedge \\ D \text{ is not } (\delta - \widetilde{\epsilon}/2)\text{-}\mathsf{Dist}^{(\mathsf{ind}+1)} \end{array} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)\right] \\ &\quad \geq \Pr\left[D \text{ is } \delta\text{-}\mathsf{Dist}^{(\mathsf{ind})} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)\right] \\ &\qquad - \Pr\left[D \text{ is } (\delta - \widetilde{\epsilon}/2)\text{-}\mathsf{Dist}^{(\mathsf{ind}+1)} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, w)\right] \\ &\quad \geq \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\delta}(\lambda, i) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind}+1)}_{\delta-\widetilde{\epsilon}/2}(\lambda, i). \end{aligned}$$

Finally substituting $\delta = \epsilon - \widetilde{\epsilon} \cdot (w + \mathsf{ind} - 1)$, we get

$$\Pr\text{-}\mathsf{Gap}^{(\mathsf{ind})}_{\mathcal{A},n,\widetilde{\epsilon}/2}(\lambda, w) \geq \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind})}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind}-1)}(\lambda, w) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{(\mathsf{ind}+1)}_{\epsilon-\widetilde{\epsilon}\cdot(w+\mathsf{ind})+\widetilde{\epsilon}/2}(\lambda, w).$$

This concludes the proof. ∎

5.2.5 Proof of Theorem 5.4

We need to show that for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$, non-negligible function $\epsilon(\cdot)$, there exists a negligible functions $\mathrm{negl}_1(\cdot), \mathrm{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, and every index $\mathsf{ind} \in [k]$,

$$\text{Pr-}\mathsf{SubTr}^{(\mathsf{ind})}_{\mathcal{A},n,\epsilon}(\lambda) \geq \frac{\left(\sum_w \text{Pr-}\mathsf{Gap}^{(\mathsf{ind})}_{\mathcal{A},n,\widetilde{\epsilon}}(\lambda, w)\right)}{n(\lambda)+k-1} - \mathrm{negl}_1(\lambda), \text{ and}$$

$$\text{Pr-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \geq 1 - \prod_{\mathsf{ind}\in[k]} \left(1 - \text{Pr-}\mathsf{SubTr}^{(\mathsf{ind})}_{\mathcal{A},n,\epsilon}(\lambda)\right) - \mathrm{negl}_2(\lambda).$$

The first inequality follows directly by applying a Chernoff bound. Note that whenever there is a gap in distinguishing advantage (between index ind and $\mathsf{ind}+1$ ciphertexts) for any position w, then with at most negligible probability the $\mathsf{Subtrace}$ algorithm outputs 1. This follows by applying a Chernoff bound similar to that used in Lemma 5.3.

For second statement, first recall the event Tr which is similar to $\mathsf{Cor\text{-}Tr}$, except that the output of the trace should be in $\{1, 2, \ldots, n\}$ (in particular, it is not required that the output be in the set S of keys queried). Now we can write the following

$$\begin{aligned}
\Pr[\mathsf{Cor\text{-}Tr}] &= \Pr[\mathsf{Tr}] - \Pr[\mathsf{Fal\text{-}Tr}] \\
&\geq \Pr[\mathsf{Tr}] - \mathrm{negl}(\lambda) \text{ (using Theorem 5.2)} \\
&\geq 1 - \Pr\left[\ \overline{\mathsf{Tr}}\ \right] - \mathrm{negl}(\lambda)
\end{aligned}$$

Also, by using independence of $\mathsf{SubTr}^{(\mathsf{ind})}$ events, we get that

$$\Pr\left[\ \overline{\mathsf{Tr}}\ \right] = \prod_{\mathsf{ind}\in[k]} \left(1 - \Pr[\mathsf{SubTr}^{(\mathsf{ind})}]\right).$$

This gives us the following

$$\Pr[\mathsf{Cor\text{-}Tr}] \geq 1 - \prod_{\mathsf{ind}\in[k]} \left(1 - \Pr[\mathsf{SubTr}^{(\mathsf{ind})}]\right) - \mathrm{negl}(\lambda).$$

This concludes the proof.

6 Construction: Mixed Bit Matching Encryption Scheme

Let $\mathsf{Grp\text{-}Gen}$ be an algorithm that takes as input security parameter 1^λ and outputs $\mathsf{params} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e(\cdot,\cdot), g_1, g_2)$ where p is a λ bit prime, $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are groups of order p, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficiently computable non-degenerate bilinear map and g_1, g_2 are generators of $\mathbb{G}_1, \mathbb{G}_2$ respectively.

$(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{mBME.Setup}(1^\lambda, 1^\ell)$: The setup algorithm first chooses $\mathsf{params} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e(\cdot,\cdot), g_1, g_2) \leftarrow \mathsf{Grp\text{-}Gen}(1^\lambda)$. It chooses $\alpha \leftarrow \mathbb{Z}_p$, $a_i \leftarrow \mathbb{Z}_p, b_i \leftarrow \mathbb{Z}_p, c_i \leftarrow \mathbb{Z}_p$ for each $i \in [\ell]$. The public key consists of params, $e(g_1, g_2)^\alpha$, $\prod_{i\in[\ell]} g_1^{a_i \cdot b_i + c_i}$ and $\{g_1^{a_i}\}_{i\in[\ell]}$, while the master secret key consists of $\left(\mathsf{params}, \alpha, \{a_i, b_i, c_i\}_{i\in\ell}\right)$.

$\mathsf{sk} \leftarrow \mathsf{mBME.KeyGen}(\mathbf{x}, \mathsf{msk})$: Let $\mathsf{msk} = \left(\mathsf{params}, \alpha, \{a_i, b_i, c_i\}_{i\in\ell}\right)$. The key generation algorithm first chooses $t \leftarrow \mathbb{Z}_p$ and $u_i \leftarrow \mathbb{Z}_p$ for each $i \in [\ell]$. It computes $K_0 = g_2^\alpha \cdot \left(\prod_{i\in[\ell]} g_2^{-t\cdot c_i}\right) \cdot \left(\prod_{i:x_i=0} g_2^{-u_i \cdot a_i}\right)$. Next, it sets $K_1 = g_2^t$, and for each $i \in [\ell]$, $K_{2,i} = g^{-t\cdot b_i}$ if $x_i = 1$, else $K_{2,i} = g_2^{-t\cdot b_i + u_i}$. The key is $\left(K_0, K_1, \{K_{2,i}\}_{i\in[\ell]}\right)$.

$\mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}SK}(m, \mathbf{y}, \mathsf{msk})$: Let $\mathsf{msk} = \big(\mathsf{params}, \alpha, \{a_i, b_i, c_i\}_{i \in \ell}\big)$. The secret key encryption algorithm first chooses $s \leftarrow \mathbb{Z}_p$, and for each $i \in [\ell]$ such that $y_i = 0$, it chooses $r_i \leftarrow \mathbb{Z}_p$. It sets $C = m \cdot e(g_1, g_2)^{\alpha \cdot s}$, $C_0 = g_1^s$, $C_1 = \left(\prod_{i:y_i=1} g_1^{s \cdot (a_i \cdot b_i + c_i)}\right) \cdot \left(\prod_{i:y_i=0} g_1^{s \cdot c_i + a_i \cdot b_i \cdot r_i}\right)$. For each $i \in [\ell]$, it sets $C_{2,i} = g_1^{a_i \cdot s}$ if $y_i = 1$, else $C_{2,i} = g_1^{a_i \cdot r_i}$ if $y_i = 0$. The ciphertext is $\left(C, C_0, C_1, \{C_{2,i}\}_{i \in [\ell]}\right)$.

$\mathsf{ct} \leftarrow \mathsf{mBME.Enc\text{-}PK}(m, \mathsf{pk})$: Let $\mathsf{pk} = (\mathsf{params}, e(g_1, g_2)^{\alpha}, \prod_{i \in \ell} g_1^{a_i \cdot b_i + c_i}, \{g_1^{a_i}\}_{i \in [\ell]})$. The public key encryption algorithm is identical to the secret key encryption algorithm. It first chooses $s \leftarrow \mathbb{Z}_p$. It sets $C = m \cdot e(g_1, g_2)^{\alpha \cdot s}$, $C_0 = g_1^s$, $C_1 = \left(\prod_{i \in \ell} g_1^{a_i \cdot b_i + c_i}\right)^s$. For each $i \in [\ell]$, it sets $C_{2,i} = (g_1^{a_i})^s$. The ciphertext is $\left(C, C_0, C_1, \{C_{2,i}\}_{i \in [\ell]}\right)$.

$z \leftarrow \mathsf{mBME.Dec}(\mathsf{ct}, \mathsf{sk})$: Let $\mathsf{ct} = \left(C, C_0, C_1, \{C_{2,i}\}_{i \in [\ell]}\right)$ and $\mathsf{sk} = \left(K_0, K_1, \{K_{2,i}\}_{i \in [\ell]}\right)$. The decryption algorithm outputs

$$\frac{C}{e(C_0, K_0) \cdot e(C_1, K_1) \cdot \prod_{i \in [\ell]} e(C_{2,i}, K_{2,i})}.$$

6.1 Correctness

Fix any security parameter λ, message m, vectors $\mathbf{x}, \mathbf{y}$ such that $f(\mathbf{x}, \mathbf{y}) = 1$ and public key $\mathsf{pk} = (\mathsf{params}, e(g_1, g_2)^{\alpha}, \prod_{i \in [\ell]} g_1^{a_i \cdot b_i + c_i}, \{g_1^{a_i}\}_{i \in [\ell]})$. Let $(s, \{r_i\}_{i:y_i=0})$ be the randomness used during encryption, $(t, \{u_i\}_{i:x_i=0})$ the randomness used during key generation, ciphertext $\mathsf{ct} = (C, C_0, C_1, \{C_{2,i}\}_{i \in [\ell]})$ and key $\mathsf{sk} = (K_0, K_1, \{K_{2,i}\}_{i \in [\ell]})$. To show that decryption works correctly, it suffices to show that $e(C_0, K_0) \cdot e(C_1, K_1) \cdot \left(\prod_{i \in [\ell]} e(C_{2,i}, K_{2,i})\right) = e(g_1, g_2)^{\alpha \cdot s}$.

$$\begin{aligned}
& e(C_0, K_0) \cdot e(C_1, K_1) \cdot \left(\prod_{i \in [\ell]} e(C_{2,i}, K_{2,i})\right) \\
&= \left(e(g_1, g_2)^{\alpha \cdot s - (\sum_i s \cdot t \cdot c_i) - (\sum_{i:x_i=0} s \cdot u_i \cdot a_i)}\right) \cdot \left(e(g_1, g_2)^{(\sum_i s \cdot t \cdot c_i) + (\sum_{i:y_i=1} s \cdot t \cdot a_i \cdot b_i) + (\sum_{i:y_i=0} t \cdot a_i \cdot b_i \cdot r_i)}\right) \\
&\quad \cdot \left(e(g_1, g_2)^{-(\sum_{i:y_i=1} t \cdot s \cdot a_i \cdot b_i) - (\sum_{i:y_i=0} t \cdot a_i \cdot b_i \cdot r_i) + (\sum_{i:x_i=0} a_i \cdot s \cdot u_i)}\right)
\end{aligned}$$

In the second step, we use the fact that since $f(\mathbf{x}, \mathbf{y}) = 1$, whenever $x_i = 0$, $y_i = 1$ (if this was not the case, then we would have, for all i such that $x_i = y_i = 0$, $e(g_1, g_2)^{u_i \cdot a_i \cdot r_i}$ terms in the product). Simplifying the expression, we get the desired product $e(g_1, g_2)^{\alpha \cdot s}$.

6.2 Security

Theorem 6.1. Assuming Assumption 1 and Assumption 2, the mixed mBME scheme described above satisfies key hiding (Definition 4.3), ciphertext hiding (Definition 4.2), and pk-sk ciphertext indistinguishability (Definition 4.1) security properties.

To prove security, we need to show that the scheme satisfies Definition 4.1, Definition 4.2 and Definition 4.3. First, it is easy to check that the scheme satisfies Definition 4.1. This is because the distribution of an encryption of m using public key pk is identical to the distribution of an encryption of m for attribute 1^{ℓ}, computed using master secret key.

6.2.1 Ciphertext Hiding Security

Lemma 6.1. Assuming Assumption 1 and Assumption 2, the mixed mBME scheme described above satisfies the ciphertext hiding security property (defined in Definition 4.2).

As shown in Section 4.2, it suffices to show that the scheme satisfies Definition 4.4 and Definition 4.5. We will prove each of these properties separately.

Claim 6.1. Assuming Assumption 1, the mixed mBME scheme described above satisfies Definition 4.4.

Proof. Suppose, on the contrary, there exists a PPT adversary $\mathcal{A}$ that breaks the simplified ciphertext hiding property of our scheme with non-negligible advantage $\epsilon(\cdot)$. We will build a PPT reduction algorithm $\mathcal{B}$ that breaks Assumption 1 with advantage $\epsilon(\cdot)$.

The reduction algorithm first receives the challenge $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, g_1^x, g_1^y, g_1^{yz}, g_2^y, g_2^z, T)$ from the challenger, where $T = g_1^{x\cdot y\cdot z}$ or $g_1^{x\cdot y\cdot z+r}$. Next, it receives the challenge vectors $\mathbf{y}, \mathbf{y}'$ from the adversary $\mathcal{A}$. Let j be the (unique) index such that $y_j \neq y'_j$, ones the set of indices such that $y_i = y'_i = 1$, and zeroes the set such that $y_i = y'_i = 0$.

The reduction algorithm must first generate the public key. Next it receives either ciphertext queries or key queries and a message m from the adversary. Then it generates the challenge ciphertext. Finally, it receives more ciphertext and key queries and the adversary's guess, which it uses to break Assumption 2.

Public key The reduction algorithm implicitly sets $a_j = y \cdot z$ and chooses $b_j, c_j \leftarrow \mathbb{Z}_p$ along with $\alpha \leftarrow \mathbb{Z}_p$. Next, for each $i \in \mathsf{ones}$, it chooses $a_i, b_i, c_i \leftarrow \mathbb{Z}_p$. For each $i \in \mathsf{zeroes}$, it chooses $a'_i, b_i, c_i \leftarrow \mathbb{Z}_p$ and implicitly sets $a_i = y + a'_i$. It computes the public key components as follows.

$$(\mathsf{pk}_{1,i}, \mathsf{pk}_{2,i}) = \begin{cases} \left(\left(g_1^{y\cdot z}\right)^{b_i} \cdot g_1^{c_i}, g_1^{y\cdot z} \right) & \text{if } i = j, \\ \left(g_1^{a_i \cdot b_i + c_i}, g_1^{a_i} \right) & \text{if } i \in \mathsf{ones}, \\ \left(\left(g_1^{y}\right)^{b_i} \cdot g_1^{a'_i \cdot b_i + c_i}, g_1^{y} \cdot g_1^{a'_i} \right) & \text{if } i \in \mathsf{zeroes}. \end{cases}$$

All the above terms can be computed using only g_1, g_1^y and $g_1^{y\cdot z}$. The public key is set to be $\mathsf{pk} = e(g_1, g_2)^{\alpha}, \prod_{i\in[\ell]} \mathsf{pk}_{1,i}, \left\{\mathsf{pk}_{2,i}\right\}_{i\in[\ell]}$.

Next, the adversary is allowed to query for polynomially many key/ciphertext queries, with the restriction that for every secret key query $\mathbf{x}$, $f(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}, \mathbf{y}')$, which are handled as follows.

Pre-challenge ciphertext queries For the ciphertext query $(m, \mathbf{w})$, the reduction algorithm chooses $q \leftarrow \mathbb{Z}_p$ and $v_i \leftarrow \mathbb{Z}_p$ for each $i \in [\ell]$. It then sets $C = m \cdot e(g_1, g_2)^{\alpha\cdot q}$, $C_0 = g_1^q$. Next, for the term C_1, note that the reduction algorithm can compute $g_1^{a_i\cdot b_i}$ and $g_1^{c_i}$ for all $i \in [\ell]$ using $g_1, g_1^y, g_1^{y\cdot z}$. As a result, for each $i \in [\ell]$, the reduction algorithm can compute both $g_1^{s\cdot(a_i\cdot b_i + c_i)}$ and $g_1^{s\cdot c_i + a_i\cdot b_i\cdot v_i}$, and hence, it can compute C_1. Finally, it can compute $C_{2,i}$ since it can compute $g_1^{a_i}$ for all $i \in [\ell]$ (using g_1, g_1^y and $g_1^{y\cdot z}$). Therefore, the ciphertext queries can be simulated perfectly.

Pre-challenge secret key queries Let $\mathbf{x}$ be the vector queried by the adversary. Here, we have two cases, depending on whether $x_j = 0$ or $x_j = 1$. In the first case, if $x_j = 1$, then the reduction algorithm chooses $t \leftarrow \mathbb{Z}_p$, and for all i such that $x_i = 0$, it chooses $u_i \leftarrow \mathbb{Z}_p$. It sets $K_1 = g_2^t$, $K_0 = \left(\prod_{i\in[\ell]} g_2^{-t\cdot c_i}\right) \cdot \left(\left(\prod_{i:x_i=0, i\in\mathsf{ones}} g_2^{-u_i\cdot a_i}\right) \cdot \left(\prod_{i:x_i=0, i\in\mathsf{zeroes}} (g_2^y)^{-u_i} \cdot g_2^{-u_i\cdot a'_i}\right)\right)$. Finally, for each $i \in [\ell]$, both $g_2^{-t\cdot b_i}$ and $g_2^{-t\cdot b_i + u_i}$ can be generated using g_2. Therefore, in this case, the key query can be handled using g_2, g_2^y.

In the second case, since $x_j = 0$, there exists an index $j^* \in [\ell]$ such that $y_{j^*} = y'_{j^*} = x_{j^*} = 0$. In this case, the reduction algorithm chooses $t \leftarrow \mathbb{Z}_p$, and chooses $u_i \leftarrow \mathbb{Z}_p$ for all i such that $i \neq j^*$ and $x_i = 0$. It chooses u'_{j^*} and sets $u_{j^*} = -z \cdot u_j + u'_{j^*}$. As in the first case, the reduction algorithm can compute $K_1 = g_2^t$. It can also compute $g_2^{-t\cdot c_i}$ for all $i \in [\ell]$, and for all i such that $x_i = 0$ and $i \notin \{j^*, j\}$, it can compute $g_2^{-u_i\cdot a_i}$ using g_2 and g_2^y. Therefore, to compute K_0, it suffices to compute $g_2^{-u_j\cdot a_j - u_{j^*}\cdot a_{j^*}}$. Plugging in the values of u_{j^*}, a_{j^*}, u_j and a_j, it follows that $g_2^{-u_j\cdot a_j - u_{j^*}\cdot a_{j^*}} = (g_2^z)^{a'_{j^*}\cdot u_j} \cdot (g_2^y)^{-u'_{j^*}} \cdot g_2^{-u'_{j^*}\cdot a'_{j^*}}$, which can be computed using g_2, g_2^y, g_2^z. Finally, the reduction algorithm needs to compute $K_{2,i}$ for each $i \in [\ell]$. For all $i \in [\ell]$, the reduction algorithm can compute $g_2^{-t \cdot b_i}$ and $g_2^{u_i}$ using g_2 and g_2^z. Therefore, the entire secret key can be computed using g_2, g_2^y, g_2^z.

After all pre-challenge key/ciphertext queries, the adversary sends a message m, which the reduction uses to compute the challenge ciphertext.

Challenge ciphertext The reduction algorithm computes the challenge ciphertext ct^*, which corresponds to an encryption of message m for attribute $\mathbf{y}$ or $\mathbf{y}'$. It implicitly sets $s = x$. For each $i \neq j$, it chooses $r_i \leftarrow \mathbb{Z}_p$, and it implicitly sets $r_j = r/(y \cdot z) + x$. It sets $C^* = m \cdot e(g_1^x, g_2)^\alpha$, $C_0^* = g_1^x$, $C_1^* = \left(\prod_{i \in \mathsf{ones}} (g_1^x)^{a_i \cdot b_i + c_i}\right) \cdot \left(\prod_{i \in \mathsf{zeroes}} (g_1^x)^{c_i} \cdot (g_1^y)^{b_i \cdot r_i} \cdot (g_1)^{a_i' \cdot b_i \cdot r_i}\right) \cdot \left(T^{b_j} \cdot (g_1^x)^{c_j}\right)$. Next, it sets the $C_{2,i}^*$ components as follows.

$$C_{2,i}^* = \begin{cases} (g_1^x)^{a_i} & \text{if } i \in \mathsf{ones}, \\ (g_1^y)^{r_i} \cdot g_1^{a_i' \cdot r_i} & \text{if } i \in \mathsf{zeroes}, \\ T \text{ if } i = j \end{cases}$$

It sends $\mathsf{ct}^* = \left(C^*, C_0^*, C_1^*, \{C_{2,i}^*\}_{i \in [\ell]}\right)$ to $\mathcal{A}$.

Next, the adversary is allowed to query for polynomially many more key/ciphertext queries, which are computed the same way as the pre-challenge queries. After all post-challenge key/ciphertext queries, the adversary sends its guess b', and the reduction algorithm forwards this guess to the challenger.

Analysis : Depending on whether $T = g_2^{x \cdot y \cdot z}$ or $T = g_2^{x \cdot y \cdot z + r}$, the reduction algorithm either outputs an encryption of m for $\mathbf{y}$ or $\mathbf{y}'$. Also, note that the ciphertext and key queries are distributed as in the security game. In particular, all the exponents $\{a_i, b_i, c_i\}$, the randomness chosen for key/ciphertext queries are uniformly random elements in $\mathbb{Z}_p$. Therefore, if $\mathcal{A}$ wins with advantage ϵ, then $\mathcal{B}$ breaks Assumption 1 with advantage ϵ.

■

Claim 6.2. Assuming Assumption 2, the mixed mBME scheme described above satisfies Definition 4.5.

Proof. Suppose, on the contrary, there exists a PPT adversary $\mathcal{A}$ that breaks the message-only ciphertext hiding property of our scheme with non-negligible advantage $\epsilon(\cdot)$. We will build a PPT reduction algorithm $\mathcal{B}$ that breaks Assumption 1 with advantage $\frac{\epsilon(\cdot)}{2}$.

The reduction algorithm first receives the challenge $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, g_1^x, g_1^y, g_2^y, g_2^z, T)$ from the challenger, where $T = g_1^{x \cdot y \cdot z}$ or $g_1^{x \cdot y \cdot z + r}$. Next, it receives the challenge vector $\mathbf{y}$ from the adversary $\mathcal{A}$. Let ones the set of indices such that $y_i = 1$ and zeroes the set such that $y_i = 0$.

The reduction algorithm must first generate the public key. Next it receives either ciphertext queries or key queries and then challenge message m_0, m_1 from the adversary. Then it generates the challenge ciphertext. Finally, it receives more ciphertext and key queries and the adversary's guess, which it uses to break Assumption 2.

Public key For each $i \in [\ell]$, the reduction algorithm picks $b_i, c_i \leftarrow \mathbb{Z}_p$. Next, for each $i \in \mathsf{ones}$, the reduction algorithm chooses $a_i \leftarrow \mathbb{Z}_p$, and for each $i \in \mathsf{zeroes}$, it choodes $a_i' \leftarrow \mathbb{Z}_p$ and implicitly sets $a_i = y + a_i'$. It also implicitly sets $\alpha = yz$. It computes the public key components as follows.

$$\left(\mathsf{pk}_{1,i}, \mathsf{pk}_{2,i}\right) = \begin{cases} \left(g_1^{a_i \cdot b_i + c_i}, g_1^{a_i}\right) & \text{if } i \in \mathsf{ones}, \\ \left((g_1^y)^{b_i} \cdot g_1^{a_i' \cdot b_i + c_i}, g_1^y \cdot g_1^{a_i'}\right) & \text{if } i \in \mathsf{zeroes}. \end{cases}$$

All the above terms can be computed using only g_1 and g_1^y.

The public key is set to be $\mathsf{pk} = e(g_1^y, g_1^z), \prod_{i \in [\ell]} \mathsf{pk}_{1,i}, \{\mathsf{pk}_{2,i}\}_{i \in [\ell]}$.

Next, the adversary is allowed to query for polynomially many key/ciphertext queries, with the restriction that for every secret key query $\mathbf{x}$, $f(\mathbf{x}, \mathbf{y}) = 0$, which are handled as follows.

Pre-challenge ciphertext queries For the ciphertext query $(m, \mathbf{w})$, the reduction algorithm chooses $q \leftarrow \mathbb{Z}_p$ and $v_i \leftarrow \mathbb{Z}_p$ for each $i \in [\ell]$. It then sets $C = m \cdot e(g_1^y, g_2^z)^q$, $C_0 = g_1^q$. Next, for the term C_1, note that the reduction algorithm can compute $g_1^{a_i \cdot b_i}$ and $g_1^{c_i}$ for all $i \in [\ell]$ using g_1 and g_1^y. As a result, for each $i \in [\ell]$, the reduction algorithm can compute both $g_1^{q\cdot(a_i \cdot b_i + c_i)}$ and $g_1^{q \cdot c_i + a_i \cdot b_i \cdot v_i}$, and hence, it can compute C_1. Finally, it can compute $C_{2,i}$ since it can compute $g_1^{a_i}$ for all $i \in [\ell]$ (using g_1 and g_1^y). Therefore, the ciphertext queries can be simulated perfectly.

Pre-challenge secret key queries Let $\mathbf{x}$ be the vector queried by the adversary. By the definition of the security game, $f(\mathbf{x}, \mathbf{y}) = 0$ so there exists an index $j^* \in [\ell]$ such that $y_{j^*} = x_{j^*} = 0$. The reduction algorithm chooses $t \leftarrow \mathbb{Z}_p$, and chooses $u_i \leftarrow \mathbb{Z}_p$ for all i such that $i \neq j^*$ and $x_i = 0$. It chooses u'_{j^*} and sets $u_{j^*} = z + u'_{j^*}$. As in the first case, the reduction algorithm can compute $K_1 = g_2^t$. It can also compute $g_2^{-t \cdot c_i}$ for all $i \in [\ell]$, and for all i such that $x_i = 0$ and $i \neq j^*$, it can compute $g_2^{-u_i \cdot a_i}$ using g_2 and g_2^y. Therefore, to compute K_0, it suffices to compute $g_2^{\alpha - u_{j^*} \cdot a_{j^*}}$. Plugging in the values of α, u_{j^*} and a_{j^*}, it follows that $g_2^{\alpha - u_{j^*} \cdot a_{j^*}} = (g_2^z)^{-a'_{j^*}} \cdot (g_2^y)^{-u'_{j^*}} \cdot g_2^{-u'_{j^*} \cdot a'_{j^*}}$, which can be computed using g_2, g_2^y, g_2^z. Finally, the reduction algorithm needs to compute $K_{2,i}$ for each $i \in [\ell]$. For all $i \in [\ell]$, the reduction algorithm can compute $g_2^{-t \cdot b_i}$ and $g_2^{u_i}$ using g_2 and g_2^z. Therefore, the entire secret key can be computed using g_2, g_2^y, g_2^z.

After all pre-challenge key/ciphertext queries, the adversary sends the challenge messages m_0, m_1, which the reduction uses to compute the challenge ciphertext.

Challenge ciphertext The reduction algorithm picks a random $b \leftarrow \{0, 1\}$ and computes the challenge ciphertext ct^*, which corresponds to an encryption of message m_b for attribute $\mathbf{y}$ or an encryption of a random element. The reduction algorithm implicitly sets $s = x$. For each $i \in \mathsf{zeroes}$, it chooses $r_i \leftarrow \mathbb{Z}_p$. It sets $C^* = m_b \cdot e(T, g_2)$, $C_0^* = g_1^x$, $C_1^* = \left(\prod_{i \in \mathsf{ones}} (g_1^x)^{a_i \cdot b_i + c_i}\right) \cdot \left(\prod_{i \in \mathsf{zeroes}} (g_1^x)^{c_i} \cdot (g_1^y)^{b_i \cdot r_i} \cdot (g_1)^{a'_i \cdot b_i \cdot r_i}\right)$. Next, it sets the $C_{2,i}^*$ components as follows.

$$C_{2,i}^* = \begin{cases} (g_1^x)^{a_i} & \text{if } i \in \mathsf{ones}, \\ (g_1^y)^{r_i} \cdot g_1^{a'_i \cdot r_i} & \text{if } i \in \mathsf{zeroes} \end{cases}$$

It sends $\mathsf{ct}^* = \left(C^*, C_0^*, C_1^*, \{C_{2,i}^*\}_{i \in [\ell]}\right)$ to $\mathcal{A}$.

Next, the adversary is allowed to query for polynomially many more key/ciphertext queries, which are computed the same way as the pre-challenge queries. After all post-challenge key/ciphertext queries, the adversary sends its guess b'. If $b' = b$, the reduction algorithm guesses 1, otherwise guesses 0.

Analysis : Depending on whether $T = g_2^{x \cdot y \cdot z}$ or $T = g_2^{x \cdot y \cdot z + r}$, the reduction algorithm either plays the honest security game with an encryption of m_b under vector $\mathbf{y}$ or sends an encryption of a random message. Also, note that the ciphertext and key queries are distributed as in the security game. In particular, all the exponents $\{a_i, b_i, c_i\}$, the randomness chosen for key/ciphertext queries are uniformly random elements in $\mathbb{Z}_p$. Therefore, if $\mathcal{A}$ wins with advantage ϵ, then $\mathcal{B}$ breaks Assumption 1 with advantage $\frac{\epsilon}{2}$.

■

6.2.2 Key-Hiding Security

Lemma 6.2. Assuming Assumption 2, the mixed mBME scheme described above satisfies the key hiding security property (defined in Definition 4.3).

In order to prove this lemma, it suffices to show that the scheme satisfies Definition 4.6 (using Theorem 4.2).

Proof. Suppose, on the contrary, there exists a PPT adversary $\mathcal{A}$ that breaks the simplified key hiding property of our scheme with non-negligible advantage $\epsilon(\cdot)$. We will build a PPT reduction algorithm $\mathcal{B}$ that breaks Assumption 2 with advantage $\epsilon(\cdot)$.

The reduction algorithm first receives the challenge $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, g_1^y, g_1^z, g_2^x, g_2^y, T)$ from the challenger, where $T = g_2^{x \cdot y \cdot z}$ or $g_2^{x \cdot y \cdot z + r}$. Next, it receives the challenge vectors $\mathbf{x}, \mathbf{x}'$ from the adversary $\mathcal{A}$. Let j be the (unique) index such that $x_j \neq x'_j$, ones the set of indices such that $x_i = x'_i = 1$, and zeroes the set such that $x_i = x'_i = 0$.

Public key The reduction algorithm implicitly sets $b_j = y \cdot z$, $c_j = -a_j \cdot y \cdot z + c'_j$. For all $i \in \mathsf{ones}$, it chooses $a_i, b_i, c_i \leftarrow \mathbb{Z}_p$. For all $i \in \mathsf{zeroes}$, it chooses a_i, b'_i, c'_i and impicitly sets $b_i = a_j \cdot y + b'_i$ and $c_i = -a_j \cdot a_i \cdot y + c'_i$.[7] In particular, it sets

$$\left(\mathsf{pk}_{1,i}, \mathsf{pk}_{2,i}\right) = \begin{cases} \left(g_1^{c'_j}, g_1^{a_j}\right) & \text{if } i = j, \\ \left(g_1^{c_i}, g_1^{a_i}\right) & \text{if } i \in \mathsf{ones}, \\ \left(g_1^{a_i \cdot b'_i + c'_i}\right) & \text{if } i \in \mathsf{zeroes}. \end{cases}$$

All the above terms can be computed using only g_1. The public key is set to be $\mathsf{pk} = \left\{\mathsf{pk}_{1,i}\right\}_{i \in [\ell]}, \left\{\mathsf{pk}_{2,i}\right\}_{i \in [\ell]}$.

Challenge secret key Next, it sends the challenge secret key (which is a secret key corresponding to either $\mathbf{x}$ or $\mathbf{x}'$). It sets $K_1^* = g_2^x$. It then chooses $\alpha \leftarrow \mathbb{Z}_p$, implicitly sets $u_j = -r$ and $u_i = a_j \cdot x \cdot y + u'_i$ for all $i \in \mathsf{zeroes}$. It sets $K_0^* = g_2^\alpha \cdot \left(\prod_{i \in \mathsf{ones}} (g_2^x)^{-c_i}\right) \cdot \left(\prod_{i \in \mathsf{zeroes}} (g_2^x)^{-c'_i} \cdot \left(g_2^{-u'_i \cdot a_i}\right)\right) \cdot \left(T^{a_j} \cdot (g_2^x)^{-c'_j}\right)$. The reduction algorithm then sets $K_{2,i}$ as follows:

$$K_{2,i}^* = \begin{cases} (g_2^x)^{-b_i} & \text{if } i \in \mathsf{ones}, \\ (g_2^x)^{-b'_i} \cdot (g_2)^{u'_i} & \text{if } i \in \mathsf{zeroes}, \\ T^{-1} & \text{if } i = j \end{cases}$$

Finally, it sends $\mathsf{sk}^* = \left(K_0^*, K_1^*, \left\{K_{2,i}^*\right\}_{i \in [\ell]}\right)$ to the adversary.

Next, the adversary is allowed to query for secret keys or ciphertexts, with the restriction that for every ciphertext query $(m, \mathbf{y})$, $f(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}', \mathbf{y})$. First, let us consider the secret key queries.

Secret key queries Let $\mathbf{w}$ denote the secret key query. The reduction algorithm chooses $q \leftarrow \mathbb{Z}_p$ and $v_i \leftarrow \mathbb{Z}_p$ for all i s.t. $w_i = 0$. It sets $K_1 = g_2^q$, $K_0 = g_2^\alpha \cdot \Big((\prod_{i \in \mathsf{ones}} g_2^{-q \cdot c_i}) \cdot \left(\prod_{i \in \mathsf{zeroes}} (g_2^y)^{-q \cdot a_i \cdot a_j} \cdot g_2^{-q \cdot c'_i}\right) \cdot \left(g_2^{x \cdot y}\right)^{q \cdot a_j} \cdot g_2^{-q \cdot c'_j} \Big) \cdot \left(\prod_{i : w_i = 0} g_2^{-v_i \cdot a_i}\right)$. Finally, it can compute $K_{2,i}$ since it can compute $g_2^{b_i}$ for all $i \in [\ell]$ (using g_2, g_2^y and $g_2^{y \cdot z}$), and the reduction algorithm knows $q, \{v_i\}_{i : w_i = 0}$. Therefore, the key queries can be simulated perfectly. The reduction algorithm can compute $K_{2,i}$ as follows. [8]

$$K_{2,i} = \begin{cases} (g_2)^{-q \cdot b_i} & \text{if } i \in \mathsf{ones}, w_i = 1, \\ (g_2)^{-q \cdot b_i + v_i} & \text{if } i \in \mathsf{ones}, w_i = 0, \\ (g_2^y)^{-q \cdot a_j} \cdot (g_2)^{-q \cdot b'_i} & \text{if } i \in \mathsf{zeroes}, w_i = 1, \\ (g_2^y)^{-q \cdot a_j} \cdot (g_2)^{-q \cdot b'_i + v_i} & \text{if } i \in \mathsf{zeroes}, w_i = 0, \\ \left(g_2^{y \cdot z}\right)^{-q} & \text{if } i = j, w_i = 1, \\ \left(g_2^{y \cdot z}\right)^{-q} \cdot (g_2)^{v_i} & \text{if } i = j, w_i = 0. \end{cases}$$

It sends $\mathsf{sk} = \left(K_0, K_1, \{K_{2,i}\}_{i \in [\ell]}\right)$ to the adversary.

[7]Can probably remove a_j from b_i, c_i and u_i.

[8]Remove this part if the previous two lines are sufficiently clear.

Ciphertext queries For ciphertext queries, we have two cases. Let $\mathbf{y}$ denote the ciphertext query attribute. In the first case, we have $y_j = 1$. In this case, the reduction algorithm chooses $s \leftarrow \mathbb{Z}_p$ and $r_i \leftarrow \mathbb{Z}_p$ for each i such that $y_i = 1$. It sets $C = m \cdot e(g_1, g_2)^{\alpha \cdot s}$, $C_0 = g_1^s$, $C_1 = \left(\prod_{i:y_i=1} g_1^{s \cdot c_i'}\right) \cdot \left(\left(\prod_{i:y_i=0, i \in \mathsf{ones}} g_2^{s \cdot c_i + a_i \cdot b_i \cdot r_i}\right) \cdot \left(\prod_{i:y_i=0, i \in \mathsf{zeroes}} (g_1^y)^{-s \cdot a_j \cdot a_i + a_j \cdot a_i \cdot r_i} \cdot g_2^{s \cdot c_i' + b_i' \cdot r_i}\right)\right)$. Here, note that since $y_j = 1$, the term C_1 does not contain $g_1^{s \cdot c_j + a_j \cdot b_j \cdot r_j}$ in the product. Finally, it sets $C_{2,i} = g_1^{s \cdot a_i}$ or $g_1^{r_i \cdot a_i}$ depending on whether $y_i = 1$ or $y_i = 0$. It sends $\left(C, C_0, C_1, \{C_{2,i}\}_{i \in [\ell]}\right)$ as the ciphertext, and the reduction algorithm thus handles the first case.

In the second case, we have $y_j = 0$. Since $f(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}', \mathbf{y})$, this implies there exists an index j^* such that $x_{j^*} = x'_{j^*} = y_{j^*} = 0$ (this implies $j^* \in \mathsf{zeroes}$). The reduction algorithm chooses $s \leftarrow \mathbb{Z}_p$ and $r_i \leftarrow \mathbb{Z}_p$ for all $i \neq j^*$. For $i = j^*$, the reduction algorithm chooses r'_{j^*} and implicitly sets $r_{j^*} = z \cdot (s - r_j)/a_{j^*} + r'_{j^*}$. It then sets $C = m \cdot e(g_1, g_2)^{\alpha \cdot s}$, $C_0 = g_1^s$. Next, note that to compute C_1, it suffices to compute $g_1^{s \cdot c_j + a_j \cdot b_j \cdot r_j} \cdot g_1^{s \cdot c_{j^*} + a_{j^*} \cdot b_{j^*} \cdot r_{j^*}}$. This is because the remaining terms in the product C_1 can be computed as in the first case. Since s is chosen by $\mathcal{B}$, it can compute $g_1^{s \cdot (c_i + a_i \cdot b_i)}$ for each $i \in [\ell]$. Similarly, for $i \neq j, j^*$, it can compute $g_1^{s \cdot c_i + a_i \cdot b_i \cdot r_i}$ using g_1 and g_1^y. By setting r_{j^*} as described above (and substituting the implicit values of $a_j, b_j, c_j, a_{j^*}, b_{j^*}, c_*$), we get that

$$g_1^{s \cdot c_j + a_j \cdot b_j \cdot r_j} \cdot g_1^{s \cdot c_{j^*} + a_{j^*} \cdot b_{j^*} \cdot r_{j^*}} = g_1^{s \cdot c_j' + s \cdot c'_{j^*} + a_{j^*} \cdot b'_{j^*} \cdot r'_{j^*}} \cdot (g_1^y)^{-s \cdot a_j \cdot a_{j^*} + a_j \cdot a_{j^*} \cdot r'_{j^*}} \cdot (g_1^z)^{b'_{j^*} \cdot (s - r_j)}$$

which can be computed using g_1, g_1^y, g_1^z. Finally, the reduction algorithm needs to compute $C_{2,i}$. For $i \neq j^*$, it sets $C_{2,i} = g_1^{a_i \cdot s}$ or $g_1^{a_i \cdot r_i}$, depending on whether $y_i = 0$ or 1. For $i = j^*$, it sets $C_{2,i} = g_1^{a_i \cdot r_i'} \cdot (g_1^z)^{a_{j^*} \cdot (s - r_{j^*})}$. This concludes the second case of ciphertext hiding.

Finally, after all key/ciphertext queries, the adversary sends its guess b, and the reduction algorithm forwards this guess to the challenger. [9]

Analysis : Depending on whether $T = g_2^{x \cdot y \cdot z}$ or $T = g_2^{x \cdot y \cdot z + r}$, the reduction algorithm either outputs a key for $\mathbf{x}$ or $\mathbf{x}'$. Also, note that the ciphertext and key queries are distributed as in the security game. In particular, all the exponents $\{a_i, b_i, c_i\}$, the randomness chosen for key/ciphertext queries are uniformly random elements in $\mathbb{Z}_p$. Therefore, if $\mathcal{A}$ wins with advantage ϵ, then $\mathcal{B}$ breaks Assumption 2 with advantage ϵ. ■

7 Performance Evaluation

We provide the performance evaluation of our risky traitor tracing scheme obtained by combining the mixed bit matching encryption scheme and the transformation to risky TT provided in Sections 6 and 5, respectively. Our performance evaluation is based on concrete measurements made using the RELIC library [rel] written in the C language.

We use the BN254 curve for pairings. It provides 126-bit security level [BGDM+10]. All running times below were measured on a server with 2.93 GHz Intel Xeon CPU and 40GB RAM. Averaged over 10000 iterations, the time taken to perform an exponentiation in the groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ is approximately 0.28 ms, 1.60 ms and 0.90 ms, respectively. The time for perform a pairing operation is around 2.22 ms. The size of elements in group $\mathbb{G}_1$ is 96 bytes.

Based on the above measurements, for risky traitor tracing with parameter k we get the ciphertext size as $(96 \cdot k + 288)$ bytes, encryption time $(0.28 \cdot k + 1.74)$ ms, and decryption time $(2.226 \cdot k + 6.66)$ ms.[10] We point out in the above evaluations we consider the KEM version of our risky traitor tracing in which the

[9] Slight inaccuracy here. We are not using the absolute value definition for the assumption/security games. So the reduction algorithm should guess either b or $1 - b$, depending on whether $x_j = 0$ or 1.

[10] In these estimations, we ignore the time to evaluate the hash function on the element in the target group $\mathbb{G}_T$ since it has an insiginicant effect on the running time.

message is encrypted using a symmetric key encryption with the hash of the first component of ciphertext $e(g_1, g_2)^{\alpha \cdot s}$ is used as the secret key. That is, the hashed value could be used as an AES key to perform message encryptions. For the basic setting of risky traitor tracing, i.e. $k = 1$, we get the ciphertext size, encryption time, and decryption time to be around 384 bytes, 2.16 ms, 8.89 ms (respectively).

8 Hardness of Differentially Private Sanitization

In this section, we show that the Dwork et al. [DNR+09] result works even if the traitor tracing scheme is f-risky secure. This, together with our construction in Section C.1, results in a hardness result with query set size $2^{O(\lambda)}$ and based on assumptions over composite order bilinear groups. First, we introduce some differential privacy related preliminaries following the notations from [KMUZ16]. Next, we describe our hardness result.

8.1 Definitions

Differentially Private Algorithms. A database $D \in \mathcal{X}^n$ is a collection of n rows $x_1, \ldots, x_n$, where each row is an element of the date universe $\mathcal{X}$. We say that two databases $D, D' \in \mathcal{X}^*$ are adjacent, denoted by $D \sim D'$, if D' can be obtained from D by the addition, removal, or substitution of a single row (i.e., they differ only on a single row). Also, for any database $D \in \mathcal{X}^n$ and index $i \in \{1, 2, \ldots, n\}$, we use D_{-i} to denote a database where the i^{th} element/row in D is set removed. At a very high level, an algorithm is said to be differentially private if its behavior on all adjacent databases is similar. The formal definition is provided below.

Definition 8.1 (Differential Privacy [DMNS06])**.** Let $A : \mathcal{X}^n \to \mathcal{S}_n$ be a randomized algorithm that takes a database as input and outputs a summary. A is (ϵ, δ)-differentially private if for every pair of adjacent databases $D, D' \in \mathcal{X}^n$ and every subset $T \subseteq \mathcal{S}_n$,

$$\Pr[A(D) \in T] \leq e^{\epsilon} \Pr[A(D') \in T] + \delta.$$

Here parameters ϵ and δ could be functions in n, the size of the database.

Accuracy of Sanitizers. Note that any algorithm A that always outputs a fixed symbol, say $\perp$, already satisfies Definition 8.1. Clearly such a summary will never be useful as the summary does not contain any information about the underlying database. Thus, we also need to specify what it means for the sanitizer to be useful. As described before, in this work we study the notion of differentially private sanitizers that give accurate answers to *statistical* queries.[11] A statistical query on data universe $\mathcal{X}$ is defined by a binary predicate $q : \mathcal{X} \to \{0, 1\}$. Let $\mathcal{Q} = \{q : \mathcal{X} \to [0, 1]\}$ be a set of statistical queries on the data universe $\mathcal{X}$. Given any $n \in \mathbb{N}$, database $D \in \mathcal{X}^n$ and query $q \in \mathcal{Q}$, let $q(D) = \dfrac{\sum_{x \in D} q(x)}{n}$.

Before we define accuracy, we would like to point out that the algorithm A might represent the summary s of a database D is any arbitrary form. Thus, to extract the answer to each query q from summary s, we require that there exists an evaluator $\mathsf{Eval} : \mathcal{S} \times \mathcal{Q} \to [0, 1]$ that takes the summary and a query, and outputs an approximate answer to that query. As in prior works, we will abuse notation and simply write $q(s)$ to denote $\mathsf{Eval}(s, q)$, i.e. the algorithm's answer to query q. At a high level, an algorithm is said to be accurate if it answers every query to within some bounded error. The formal definition follows.

Definition 8.2 (Accuracy)**.** For a set $\mathcal{Q}$ of statistical queries on $\mathcal{X}$, a database $D \in \mathcal{X}^n$ and a summary $s \in \mathcal{S}$, we say that s is α-accurate for $\mathcal{Q}$ on D if

$$\forall q \in \mathcal{Q}, |q(D) - q(s)| \leq \alpha.$$

[11] Statistical queries are also referred as counting queries, predicate queries, or linear queries in the literature.

A randomized algorithm $A : \mathcal{X}^n \to \mathcal{S}$ is said to be an (α, β)-accurate sanitizer if for every database $D \in \mathcal{X}^n$,

$$\Pr_{A\text{'s coins}}[A(D) \text{ is } \alpha\text{-accurate for } \mathcal{Q} \text{ on } D] \geq 1 - \beta.$$

The parameters α and β could be functions in n, the size of the database.

Efficiency of Sanitizers. In this work, we are interested in asymptotic efficiency, thus we introduce a computation parameter $\lambda \in \mathbb{N}$. The data universe and query space, both will be parameterized by λ; that is, for every $\lambda \in \mathbb{N}$, we have a data universe $\mathcal{X}_\lambda$ and a query space $\mathcal{Q}_\lambda$. The size of databases will be bounded by $n = n(\lambda)$, where $n(\cdot)$ is a polynomial. Now the algorithm A takes as input a database $\mathcal{X}_\lambda^n$ and output a summary in $\mathcal{S}_\lambda$, where $\{\mathcal{S}_\lambda\}_{\lambda \in \mathbb{N}}$ is a sequence of output ranges. And, there is an associated evaluator Eval that takes a query $q \in \mathcal{Q}_\lambda$ and a summary $S \in \mathcal{S}_\lambda$ and outputs a real-valued answer. The definitions of differential privacy and accuracy readily extend to such sequences.

Definition 8.3 (Efficiency)**.** A sanitizer A is efficient if, on input a database $D \in \mathcal{X}_\lambda^n$, A runs in time $\mathsf{poly}(\lambda, \log(|X_\lambda|), \log(|Q_\lambda|))$, as well as on input a summary $s \in \mathcal{S}_\lambda$ and query $q \in \mathcal{Q}_\lambda$, the associated evaluator Eval runs in time $\mathsf{poly}(\lambda, \log(|X_\lambda|), \log(|Q_\lambda|))$.

8.2 Hardness of Efficient Differentially Private Sanitization from Risky Traitor Tracing

In this section, we prove hardness of efficient differentially private sanitization from risky traitor tracing schemes. The proof is an adaptation of the proofs in [DNR+09, Ull13, KMUZ16] to this restricted notion. At a high level, the idea is to set the data universe to the secret key space and each query will be associated with a ciphertext such that answer to a query on any secret key will correspond to the output of decryption of associated ciphertext using the secret key. Now to show hardness of sanitization we will prove by contradiction. The main idea is that if there exists an efficient (accurate) sanitizer, then that could be successfully used as a pirate box in the traitor tracing scheme. Next, assuming that the sanitizer satisfies differential privacy, we can argue that the sanitizer could still be a useful pirate box even if one of keys in the database is deleted, however the tracing algorithm will still output the missing key as a traitor with non-negligible probability, thereby contradicting the property that the tracing algorithm incorrectly traces with only negligible probability.

Below we state the formal theorem and give a proof. Later we also show to get a stronger hardness result if the underlying risky traitor tracing schemes also satisfies "singular trace" property (Definition 3.5).

8.2.1 Hardness from Risky Traitor Tracing

Theorem 8.1. If there exists a f-risky secure private-key no-query traitor tracing scheme $\mathcal{T} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$ (Definition 3.7), then there exists a data universe and query family $\{\mathcal{X}_\lambda, \mathcal{Q}_\lambda\}_\lambda$ such that there does not any sanitizer $A : \mathcal{X}_\lambda^n \to \mathcal{S}_\lambda$ that is simultaneously — (1) (ϵ, δ)-differentially private, (2) (α, β)-accurate for query space $\mathcal{Q}_\lambda$ on $\mathcal{X}_\lambda^n$, and (3) computationally efficient — for any $\epsilon = O(\log \lambda), \alpha < 1/2, \beta = o(1)$ and $\delta \leq f \cdot (1 - \beta)/4n$.

Proof. Let $\mathcal{T} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$ be a traitor tracing scheme with key space $\{\mathcal{K}_\lambda\}_\lambda$, message space $\{0, 1\}$ and ciphertext space $\{\mathcal{C}_\lambda\}_\lambda$. For any $\lambda \in \mathbb{N}$, the data universe is set to be $\mathcal{X}_\lambda = \mathcal{K}_\lambda$, and the distribution on databases is defined as $\mathcal{X}_\lambda^n = \left\{D \ : (\mathsf{msk}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)) \leftarrow \mathsf{Setup}(1^\lambda, 1^n), D = (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)\right\}$. In the sequel, to sample the database, we will simply write $(\mathsf{msk}, D) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$. Each query in the query space is associated with a ciphertext ct, and the output of any query q_{ct} corresponds to the decryption of associated ciphertext using the input secret key. Formally, $\mathcal{Q}_\lambda = \{\mathsf{Dec}(\cdot, \mathsf{ct}) \ : \mathsf{ct} \in \mathcal{C}_\lambda\}$, i.e. for every $q_{\mathsf{ct}} \in \mathcal{Q}_\lambda$, $q_{\mathsf{ct}}(\mathsf{sk}) = \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$.

Let A be a computationally efficient algorithm such that it is (ϵ, δ)-differentially private and (α, β)-accurate for query space $\mathcal{Q}_\lambda$ on $\mathcal{X}_\lambda^n$. From (α, β)-accuracy of A we can write that for every $(\mathsf{msk}, D) \leftarrow$

$\mathsf{Setup}(1^\lambda, 1^n)$ and every ciphertext $\mathsf{ct} \in \mathcal{C}_\lambda$, the following holds

$$\Pr_{A\text{'s coins}} [|q_{\mathsf{ct}}(A(D)) - q_{\mathsf{ct}}(D)| \le \alpha] \ge 1 - \beta. \tag{1}$$

Now from the correctness property of traitor tracing scheme, we know that for every ciphertext $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{msk}, b)$, $q_{\mathsf{ct}}(D) = b$. This is because $q_{\mathsf{ct}}(D) = \left(\sum_{\mathsf{sk} \in D} q_{\mathsf{ct}}(\mathsf{sk})\right)/n$ and for every sk, $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = b$. Also, since $\alpha < 1/2$ we can conclude that for every $(\mathsf{msk}, D) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$ and every ciphertext $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{msk}, b)$, the following holds

$$\Pr_{A\text{'s coins}} [\lceil \mathsf{Eval}(A(D), q_{\mathsf{ct}}) \rfloor = b] \ge 1 - \beta. \tag{2}$$

Consider an adversary $\mathcal{B}$ that plays the f-risky ind-secure tracing game with scheme $\mathcal{T}$. Adversary $\mathcal{B}$ runs as follows. During key query phase, $\mathcal{B}$ queries the tracing scheme challenger for all n secret keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_n$. Next, it runs the sanitizer A on all n keys with uniformly random coins. In other words, it generates a summary s as $s \leftarrow A(D)$ where $D = (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)$. Finally, $\mathcal{B}$ outputs $\mathsf{Eval}(s, \cdot)$ as the pirate decoding box, i.e. the evaluation algorithm with summary s hardwired. Note that $\mathcal{B}$ is efficient because A is efficient.

Using the previous equation, we can conlude that adversary $\mathcal{B}$ outputs a "good decoder" with probability at least $1 - \beta$. Formally, we can write that for every non-negligible function ϵ,

$$\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathcal{T}}_{\mathcal{B},n,\epsilon}(\lambda) \ge 1 - \beta. \tag{3}$$

Now since the scheme $\mathcal{T}$ is f-risky secure where $f = f(n, \lambda)$, we also get that

$$\Pr\text{-}\mathsf{Cor\text{-}Tr}^{\mathcal{T}}_{\mathcal{B},n,\epsilon}(\lambda) \ge f \cdot (1 - \beta) - \mathrm{negl}(\lambda), \tag{4}$$

where negl is a negligible function. Since $\mathcal{B}$ queries for all n keys, thus whenever there is a "correct trace" in this scenario, the trace algorithm outputs an index in $[n]$. Therefore, we can write that

$$\Pr_{\substack{(\mathsf{msk}, D) \leftarrow \mathsf{Setup}(1^\lambda, 1^n) \\ s \leftarrow A(D) \text{ and } \mathsf{Trace}\text{'s coins}}} \left[\mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk}) \in [n]\right] \ge \Pr\text{-}\mathsf{Cor\text{-}Tr}^{\mathcal{T}}_{\mathcal{B},n,\epsilon}(\lambda) \ge f \cdot (1 - \beta) - \mathrm{negl}(\lambda). \tag{5}$$

Next, we can claim that there exists an index $i^* \in [n]$ such that the trace algorithm, given $\mathsf{Eval}(s, \cdot)$ as the pirate box, outputs index i^* with probability at least $f \cdot (1 - \beta)/n$, as otherwise it would contradict the previous lower bound. Formally, we can say that there exists $i^* \in [n]$ such that

$$\Pr_{\substack{(\mathsf{msk}, D) \leftarrow \mathsf{Setup}(1^\lambda, 1^n) \\ s \leftarrow A(D) \text{ and } \mathsf{Trace}\text{'s coins}}} \left[\mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk}) = i^*\right] \ge \frac{f \cdot (1 - \beta)}{n} - \mathrm{negl}(\lambda). \tag{6}$$

Let $S_{\mathsf{msk}, D, i^*} \subseteq \mathcal{S}_\lambda$ be the set of summaries such that for every summary s in that set, the probability of trace algorithm outputting index i^* given $\mathsf{Eval}(s, \cdot)$ as the pirate box is at least f/n^2. Formally, for a given msk, D, i^*,

$$S_{\mathsf{msk}, D, i^*} := \left\{ s : \Pr_{\mathsf{Trace}\text{'s coins}} \left[\mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk}) = i^*\right] \ge \frac{f}{n^2} \right\}. \tag{7}$$

Now, using the previous two equations, we could claim the following -

$$\Pr_{\substack{(\mathsf{msk}, D) \leftarrow \mathsf{Setup}(1^\lambda, 1^n) \\ A\text{'s coins}}} [A(D) \in S_{\mathsf{msk}, D, i^*}] \ge \frac{f \cdot (1 - \beta)}{2n}. \tag{8}$$

By (ϵ, δ)-differential privacy of A, we have that

$$\Pr_{\substack{(\mathsf{msk}, D) \leftarrow \mathsf{Setup}(1^\lambda, 1^n) \\ A\text{'s coins}}} [A(D_{-i^*}) \in S_{\mathsf{msk}, D, i^*}] \ge e^{-\epsilon} \left(\frac{f \cdot (1 - \beta)}{2n} - \delta \right) \ge \frac{e^{-\epsilon} \cdot f \cdot (1 - \beta)}{4n}. \tag{9}$$

Finally, combining above equation with the definition of set S_{msk,D,i^*}, we get that

$$\Pr_{\substack{(\mathsf{msk},D)\leftarrow\mathsf{Setup}(1^\lambda,1^n)\\ s\leftarrow A(D_{-i^*})\text{ and }\mathsf{Trace}\text{'s coins}}}\left[\mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk})=i^*\right]\geq\frac{e^{-\epsilon}\cdot f\cdot(1-\beta)}{4n}\times\frac{f}{n^2}. \tag{10}$$

Now this violates the f-risky security of the traitor tracing scheme. Concretely, consider an adversary $\mathcal{B}^*$ that runs as follows. During key query phase, $\mathcal{B}^*$ queries the tracing scheme challenger for all but i^{*th} secret key, i.e., $\{\mathsf{sk}_i\}_{i\neq i^*}$. Next, it runs the sanitizer A on these $n-1$ keys with uniformly random coins, i.e. it generates a summary s as $s\leftarrow A(D_{-i^*})$ where $D=(\mathsf{sk}_1,\ldots,\mathsf{sk}_n)$. Finally, $\mathcal{B}^*$ outputs $\mathsf{Eval}(s,\cdot)$ as the pirate decoding box, i.e. the evaluation algorithm with summary s hardwired. Note that $\mathcal{B}^*$ is efficient because A is efficient. Now using the previous equation we can conclude that

$$\Pr\text{-}\mathsf{Fal}\text{-}\mathsf{Tr}^{\mathcal{T}}_{\mathcal{B}^*,n,1/2}(\lambda)\geq\frac{e^{-\epsilon}\cdot f^2\cdot(1-\beta)}{4n^3}. \tag{11}$$

In other words, the probability $\mathcal{B}^*$ leads to a "faulty trace" is non-negligible. However, since $\mathcal{T}$ is f-risky secure, the probability of a faulty trace should be negligible in the security parameter. Thus, this leads to a contradiction completing the proof. ∎

8.2.2 Hardness from Risky Traitor Tracing with Singular Trace

Theorem 8.2. If there exists a f-risky secure private-key no-query traitor tracing scheme $\mathcal{T}=(\mathsf{Setup},\mathsf{Enc},\mathsf{Dec},\mathsf{Trace})$ (Definition 3.7) satisfying singular trace property (Definition 3.5), then there exists a data universe and query family $\{\mathcal{X}_\lambda,\mathcal{Q}_\lambda\}_\lambda$ such that there does not any sanitizer $A:\mathcal{X}_\lambda^n\to\mathcal{S}_\lambda$ that is simultaneously — (1) (ϵ,δ)-differentially private, (2) (α,β)-accurate for query space $\mathcal{Q}_\lambda$ on $\mathcal{X}_\lambda^n$, and (3) computationally efficient — for any $\epsilon=O(\log\lambda),\alpha<1/2,\beta=o(1)$ and $\delta\leq f\cdot(1-\beta)/4$.

Proof. The proof of this theorem is similar to that of Theorem 8.1, therefore we only highlight the equations/components that are modified. First, the database, query space and input spaces are all identical. Thus, the proof is identical until Equation (4). Next, since the traitor tracing scheme $\mathcal{T}$ is f-risky secure as well as satisfies the singular trace property, thus we could directly conclude that there exists $i^*\in[n]$ such that

$$\Pr_{\substack{(\mathsf{msk},D)\leftarrow\mathsf{Setup}(1^\lambda,1^n)\\ s\leftarrow A(D)\text{ and }\mathsf{Trace}\text{'s coins}}}\left[\mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk})=i^*\right]\geq f\cdot(1-\beta)-\mathrm{negl}(\lambda). \tag{12}$$

In other words, we can avoid a $1/n$ loss due to the singular trace property. The remaining proof is almost identical. The only modification is that all the lower bounds in Equations 7, 8 and 9 get tighter by a factor of n, i.e. the degree of n in the denominator in all of them can be reduced by 1. With these modifications we can conclude that

$$\Pr_{\substack{(\mathsf{msk},D)\leftarrow\mathsf{Setup}(1^\lambda,1^n)\\ s\leftarrow A(D_{-i^*})\text{ and }\mathsf{Trace}\text{'s coins}}}\left[\mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk})=i^*\right]\geq\frac{e^{-\epsilon}\cdot f\cdot(1-\beta)}{4}\times\frac{f}{n}. \tag{13}$$

And, finally if we consider the same adversary $\mathcal{B}^*$ as in previous proof, then we can conclude that

$$\Pr\text{-}\mathsf{Fal}\text{-}\mathsf{Tr}^{\mathcal{T}}_{\mathcal{B}^*,n,1/2}(\lambda)\geq\frac{e^{-\epsilon}\cdot f^2\cdot(1-\beta)}{4n}. \tag{14}$$

Thus, this leads to a contradiction completing the proof. ∎

8.2.3 Hardness from Assumptions over Bilinear Groups

Combining Theorem 8.2 with Theorems 5.2, 5.3, 5.4, and 6.1, we get the following corollary.

Corollary 8.1. If Assumption 1 and Assumption 2 hold, then there exists a data universe and query family $\{\mathcal{X}_\lambda, \mathcal{Q}_\lambda\}_\lambda$ such that there does not any sanitizer $A : \mathcal{X}_\lambda^n \to \mathcal{S}_\lambda$ that is simultaneously — (1) (ϵ, δ)-differentially private, (2) (α, β)-accurate for query space $\mathcal{Q}_\lambda$ on $\mathcal{X}_\lambda^n$, and (3) computationally efficient — for any $\epsilon = O(\log \lambda), \alpha < 1/2, \beta = o(1)$ and $\delta \leq (1-\beta)/4n$.

Similary, combining Theorem 8.2 with Theorems C.2, C.3 and C.4, we get the following corollary.

Corollary 8.2. Assuming subgroup decision (Assumption 3) and subgroup hiding in target group assumptions (Assumption 4), there exists a data universe and query family $\{\mathcal{X}_\lambda, \mathcal{Q}_\lambda\}_\lambda$ such that there does not any sanitizer $A : \mathcal{X}_\lambda^n \to \mathcal{S}_\lambda$ that is simultaneously — (1) (ϵ, δ)-differentially private, (2) (α, β)-accurate for query space $\mathcal{Q}_\lambda$ on $\mathcal{X}_\lambda^n$, and (3) computationally efficient — for any $\epsilon = O(\log \lambda), \alpha < 1/2, \beta = o(1)$ and $\delta \leq (1-\beta)/4n$.

9 Amplifying the Trace Success Probability

In this section, we will show a generic transformation to amplify any traitor tracing scheme's success probability. In particular, given two traitor tracing schemes, one being f_A-risky and the other one being f_B-risky, we show how to combine them to obtain an $(f_A + f_B - f_A \cdot f_B)$-risky traitor tracing scheme. We will focus on public key traitor tracing schemes; our transformation can also be applied to private-key traitor tracing schemes.

9.1 Construction

Let $\mathcal{T}_A = (\mathsf{Setup}_A, \mathsf{Enc}_A, \mathsf{Dec}_A, \mathsf{Trace}_A)$ be an f_A-risky secure traitor tracing scheme for message space $\mathcal{M}$, and $\mathcal{T}_B = (\mathsf{Setup}_B, \mathsf{Enc}_B, \mathsf{Dec}_B, \mathsf{Trace}_B)$ an f_B-risky secure traitor tracing scheme for $\mathcal{M}$. We will now describe a new traitor tracing scheme $\mathcal{T} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Trace})$ for message space $\mathcal{M}$.

$\mathsf{Setup}(1^\lambda, 1^n)$: The setup algorithm chooses $(\mathsf{mpk}_A, \mathsf{msk}_A, (\mathsf{sk}_{A,1}, \ldots, \mathsf{sk}_{A,n})) \leftarrow \mathsf{Setup}_A(1^\lambda, 1^n)$ and $(\mathsf{mpk}_B, \mathsf{msk}_B, (\mathsf{sk}_{B,1}, \ldots, \mathsf{sk}_{B,n})) \leftarrow \mathsf{Setup}_B(1^\lambda, 1^n)$. It sets $\mathsf{mpk} = (\mathsf{mpk}_A, \mathsf{mpk}_B)$, $\mathsf{msk} = (\mathsf{msk}_A, \mathsf{msk}_B)$ and for $j \in \{1, 2, \ldots, n\}$, $\mathsf{sk}_j = (\mathsf{sk}_{A,j}, \mathsf{sk}_{B,j})$.

$\mathsf{Enc}(\mathsf{mpk}, m)$: Let $\mathsf{mpk} = (\mathsf{mpk}_A, \mathsf{mpk}_B)$. The encryption algorithm chooses $r \leftarrow \mathcal{M}$, computes $\mathsf{ct}_A \leftarrow \mathsf{Enc}_A(\mathsf{mpk}_A, m \oplus r)$ and $\mathsf{ct}_B \leftarrow \mathsf{Enc}_B(\mathsf{mpk}_B, r)$. It sets $\mathsf{ct} = (\mathsf{ct}_A, \mathsf{ct}_B)$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: Let $\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$ and $\mathsf{ct} = (\mathsf{ct}_A, \mathsf{ct}_B)$. The decryption algorithm computes $x_A \leftarrow \mathsf{Dec}_A(\mathsf{sk}_A, \mathsf{ct}_A)$, $x_B \leftarrow \mathsf{Dec}_B(\mathsf{sk}_B, \mathsf{ct}_B)$ and outputs $x_A \oplus x_B$.

$\mathsf{Trace}^D(\mathsf{msk}, 1^y, m_0, m_1)$: Let $\mathsf{msk} = (\mathsf{msk}_A, \mathsf{msk}_B)$ and $\epsilon = 1/y$. Consider the routines $\mathsf{Test\text{-}Good\text{-}}A$ and $\mathsf{Test\text{-}Good\text{-}}B$ (defined in Figure 11 and Figure 12) which take as input $r \in \mathcal{M}$ and a ciphertext, and outputs 0/1.

Routine $\mathsf{Test\text{-}Good\text{-}}A(r, \mathsf{ct}_B)$

Inputs: $r \in \mathcal{M}$, ciphertext ct_B
Output: 0/1

```
1. Set $\mathsf{count} = 0$ and $z = \lambda \cdot n/\epsilon$. For $j = 1$ to $z$, do the following:
    (a) Choose $b \leftarrow \{0,1\}$, compute $\mathsf{ct}_{A,j} \leftarrow \mathsf{Enc}_A(\mathsf{mpk}_A, m_b \oplus r)$. If $D(\mathsf{ct}_{A,j}, \mathsf{ct}_B) = b$, $\mathsf{count} = \mathsf{count}+1$.
  If $\mathsf{count}/z > 1/2 + \epsilon/3$, output 1, else output 0.
```

Figure 11: Routine $\mathsf{Test\text{-}Good\text{-}}A(r, \mathsf{ct}_B)$

Routine $\mathsf{Test\text{-}Good\text{-}}B(r, \mathsf{ct}_A)$

Inputs: $r \in \mathcal{M}$, ciphertext ct_A
Output: 0/1

```
1. Set $\mathsf{count} = 0$ and $z = \lambda \cdot n/\epsilon$. For $j = 1$ to $z$, do the following:
   (a) Choose $b \leftarrow \{0,1\}$, compute $\mathsf{ct}_{B,j} \leftarrow \mathsf{Enc}_B(\mathsf{mpk}_B, m_b \oplus r)$. If $D(\mathsf{ct}_A, \mathsf{ct}_{B,j}) = b$, $\mathsf{count} = \mathsf{count}+1$.
   If $\mathsf{count}/z > 1/2 + \epsilon/3$, output 1, else output 0.
```

Figure 12: Routine $\mathsf{Test\text{-}Good\text{-}}B(r, \mathsf{ct}_A)$

The above routines will be useful for building a pirate decoder for Trace_A and Trace_B respectively. The trace algorithm first builds a pirate decoder for Trace_A and uses Trace_A to trace a traitor. If Trace_A returns an index $i \in \{1, 2, \ldots, n\}$, then the algorithm outputs i. Else, it constructs a different pirate decoder for Trace_B and uses Trace_B to trace a traitor.

Trace attempt using Trace_A :

1. The trace algorithm first searches for an $r_A \in \mathcal{M}$ and ciphertext ct_B such that $\mathsf{Test\text{-}Good\text{-}}A(r_A, \mathsf{ct}_B) = 1$. Let $r_A = \perp$, $\mathsf{ct}_B = \perp$.
   For $i = 1$ to $\lambda \cdot n/\epsilon$, it chooses $r^i \leftarrow \mathcal{M}$, sets $\mathsf{ct}^i \leftarrow \mathsf{Enc}_B(\mathsf{mpk}_B, r^i)$ and checks if $\mathsf{Test\text{-}Good\text{-}}A(r^i, \mathsf{ct}^i) = 1$. If so, it sets $r_A = r^i$, $\mathsf{ct}_B = \mathsf{ct}^i$ and exits loop.
   If $r_A = \perp$, then quit trace attempt using Trace_A.
2. Consider the following pirate decoder D_A for $\mathcal{T}_A$. The decoder has ct_B hardwired. On input ciphertext ct_A, it outputs $D(\mathsf{ct}_A, \mathsf{ct}_B)$. The trace algorithm sets $m_{A,0} = m_0 \oplus r_A$, $m_{A,1} = m_1 \oplus r_A$ and computes $z \leftarrow \mathsf{Trace}_A^{D_A}(\mathsf{msk}_A, 1^{4y}, m_{A,0}, m_{A,1})$.
3. If $z \neq \perp$, output z. Else, perform trace attempt using Trace_B.

Trace attempt using Trace_B : This is similar to the trace attempt using Trace_B, except that the trace algorithm now builds a pirate decoding box for Trace_B.

1. The trace algorithm searches for an $r_B \in \mathcal{M}$ and ciphertext ct_A such that $\mathsf{Test\text{-}Good\text{-}}B(r_B, \mathsf{ct}_A) = 1$. Let $r_B = \perp$, $\mathsf{ct}_A = \perp$.
   For $i = 1$ to $\lambda \cdot n/\epsilon$, it chooses $r^i \leftarrow \mathcal{M}$, sets $\mathsf{ct}^i \leftarrow \mathsf{Enc}_A(\mathsf{mpk}_A, r^i)$ and checks if $\mathsf{Test\text{-}Good\text{-}}B(r^i, \mathsf{ct}^i) = 1$. If so, it sets $r_B = r^i$, $\mathsf{ct}_A = \mathsf{ct}^i$ and exits loop.
   If $r_B = \perp$, then quit trace attempt using Trace_B.
2. Consider the following pirate decoder D_B for $\mathcal{T}_B$. The decoder has ct_A hardwired. On input ciphertext ct_B, it outputs $D(\mathsf{ct}_A, \mathsf{ct}_B)$. The trace algorithm sets $m_{B,0} = m_0 \oplus r_B$, $m_{B,1} = m_1 \oplus r_B$ and computes $z \leftarrow \mathsf{Trace}_B^{D_B}(\mathsf{msk}_B, 1^{4y}, m_{B,0}, m_{B,1})$.
3. If $z \neq \perp$, output z.

Correctness The correctness of this scheme follows from the correctness of schemes $\mathcal{T}_\mathcal{A}$ and $\mathcal{T}_B$.

9.2 Security

In this section, we will show that our scheme is $\mathsf{IND\text{-}CPA}$ and $(f_A + f_B - f_A \cdot f_B)$-risky secure.

$\mathsf{IND\text{-}CPA}$ Security

Lemma 9.1. Assuming $\mathcal{T}_A$ is $\mathsf{IND\text{-}CPA}$ secure, $\mathcal{T}$ is also $\mathsf{IND\text{-}CPA}$ secure.

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ such that $\Pr[1 \leftarrow \mathsf{Expt\text{-}IND\text{-}CPA}_{\mathcal{T},\mathcal{A}}(1^\lambda, 1^n)] - 1/2 = \eta$, where η is non-negligible. We will construct a reduction algorithm $\mathcal{B}$ such that $\Pr[1 \leftarrow \mathsf{Expt\text{-}IND\text{-}CPA}_{\mathcal{T}_A,\mathcal{B}}(1^\lambda, 1^n)] - 1/2 = \eta$.

The reduction algorithm receives mpk_A from the $\mathsf{IND\text{-}CPA}$ challenger. It then chooses $(\mathsf{msk}_B,\ \mathsf{msk}_B,\ (\mathsf{sk}_{B,1},\ \ldots,\ \mathsf{sk}_{B,n})) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, sets $\mathsf{mpk} = (\mathsf{mpk}_A, \mathsf{mpk}_B)$ and sends it to $\mathcal{A}$. The adversary $\mathcal{A}$ then sends two messages m_0, m_1. The reduction algorithm chooses $r \leftarrow \mathcal{M}$. It sets $m_{0,A} = m_0 \oplus r$, $m_{1,A} = m_1 \oplus r$ and sends $(m_{0,A}, m_{1,A})$ to the challenger. The reduction algorithm receives ct_A from the challenger. It computes $\mathsf{ct}_B \leftarrow \mathsf{Enc}_B(\mathsf{mpk}_B, r)$ and sends $\mathsf{ct} = (\mathsf{ct}_A, \mathsf{ct}_B)$ to $\mathcal{A}$. The attacker sends its guess b', and the reduction algorithm forwards it to the challenger.

If the adversary's guess is correct, then so is the reduction algorithm's guess. Therefore, $\mathcal{B}$ breaks the $\mathsf{IND\text{-}CPA}$ security of $\mathcal{T}_A$ with advantage η. ■

False-Trace Probability

Lemma 9.2. Assuming $\mathcal{T}_A$ is an f_A-risky secure traitor tracing scheme, and $\mathcal{T}_B$ is an f_B-risky secure traitor tracing scheme, for every PPT adversary $\mathcal{A}$, polynomials $p(\cdot)$, $n(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \leq \mathrm{negl}(\lambda)$.

Proof. Let S denote the set of key queries made by the adversary. We define the following events :

- $\mathcal{F}_1$: Trace_A outputs an index $i \in \{1, 2, \ldots, n\} \setminus S$
- $\mathcal{F}_2$: Trace_A outputs $\perp$ and Trace_B outputs an index $i \in \{1, 2, \ldots, n\} \setminus S$

Clearly, $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon} = \Pr[\mathcal{F}_1] + \Pr[F_2]$. We will show an upper bound on $\Pr[\mathcal{F}_1]$ and $\Pr[\mathcal{F}_2]$ using the security of $\mathcal{T}_A$ and $\mathcal{T}_B$ respectively.

Claim 9.1. Assuming $\mathcal{T}_A$ is an f_A-risky secure traitor tracing scheme, for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}_1(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr[\mathcal{F}_1] \leq \mathrm{negl}_1(\lambda)$.

Proof. Suppose, on the contrary, there exists a PPT adversary $\mathcal{A}$, polynomial $n(\cdot)$ and non-negligible functions ϵ, η such that $\Pr[\mathcal{F}_1] \geq \eta(\lambda)$. We will show that there exists a non-negligible function ϵ_A and a PPT reduction algorithm $\mathcal{B}$ that queries for set S, outputs a decoding box D_A and two messages $m_{A,0}, m_{A,1}$ such that $\Pr[\mathsf{Trace}_A^{D_A}(\mathsf{msk}_A, 1^{1/\epsilon_A}, m_{A,0}, m_{A,1}) \in \{1, 2, \ldots, n\} \setminus S] \geq \eta(\lambda)$.

The reduction algorithm $\mathcal{B}$ first receives mpk_A from the challenger. It chooses $(\mathsf{mpk}_B, (\mathsf{sk}_{B,1}, \ldots, \mathsf{sk}_{B,n})) \leftarrow \mathsf{Setup}_B(1^\lambda, 1^n)$ and sends $\mathsf{mpk} = (\mathsf{mpk}_A, \mathsf{mpk}_B)$ to $\mathcal{A}$.

Next, it receives secret key queries from $\mathcal{A}$. For query corresponding to index i, the reduction algorithm sends the same query to challenger. It receives $\mathsf{sk}_{A,i}$, and it sends $\mathsf{sk}_i = (\mathsf{sk}_{A,i}, \mathsf{sk}_{B,i})$ to $\mathcal{A}$.

Finally, after all the queries, the adversary $\mathcal{A}$ sends a pirate decoding box D together with messages m_0, m_1. The reduction algorithm sets $T = \lambda \cdot n/\epsilon$. For $i = 1$ to T, it chooses $r^i \leftarrow \mathcal{M}$, computes $\mathsf{ct}_B^i \leftarrow \mathsf{Enc}_B(\mathsf{mpk}_B, r^i)$ and checks if $\mathsf{Test\text{-}Good\text{-}}A(r^i, \mathsf{ct}_B) = 1$. If no pair exists, it outputs an empty decoding box. Else, let (r_A, ct_B) be the first such pair. The reduction algorithm uses (r_A, ct_B) and decoder box D to define D_A and $m_{A,0}, m_{A,1}$. It sets $m_{A,0} = m_0 \oplus r_A$ and $m_{A,1} = m_1 \oplus r_A$. The pirate box D_A has ct_B hardwired, and it takes as input a ciphertext ct and outputs $D((\mathsf{ct}, \mathsf{ct}_B))$. $\mathcal{B}$ sends $D_A, m_{A,0}, m_{A,1}$ to the challenger.

Now, if $\Pr[\mathcal{F}_1] \geq \eta(\lambda)$, then $\Pr[\mathsf{Trace}_A^{D_A}(\mathsf{msk}_A, 1^{4/\epsilon}, m_{A,0}, m_{A,1}) \in \{1, 2, \ldots, n\} \setminus S] \geq \eta(\lambda)$.

■

Claim 9.2. Assuming $\mathcal{T}_B$ is an f_B-risky secure traitor tracing scheme, for every PPT adversary $\mathcal{A}$, polynomials $p(\cdot)$, $n(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}_1(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr[\mathcal{F}_2] \leq \text{negl}_1(\lambda)$.

The proof of this claim is identical to the previous one. Here, the reduction algorithm gets $\mathcal{T}_B$ parameters from the challenger, and generates the $\mathcal{T}_A$ parameters by itself. On receiving the decoding box D, it first tries Trace_A by itself. If this trace works, it quits. Else, it computes the box D_B and messages $m_{B,0}, m_{B,1}$ and sends them to the challenger.

■

Correct-Trace Probability First, we need to define some probabilistic events. In order to do so, let us recall the security game for traitor tracing.

1. Challenger chooses $(\mathsf{mpk} = (\mathsf{mpk}_A, \mathsf{mpk}_B),\ \mathsf{msk} = (\mathsf{msk}_A, \mathsf{msk}_B),\ (\mathsf{sk}_1 = (\mathsf{sk}_{A,1}, \mathsf{sk}_{B,1}),\ \ldots,\ \mathsf{sk}_n = (\mathsf{sk}_{A,n}, \mathsf{sk}_{B,n}))) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$ and sends mpk to $\mathcal{A}$.

2. Adversary queries for secret keys. For each queried index $i \in \{1, 2, \ldots, n\}$, the challenger sends sk_i.

3. Let S denote the set of keys queried by $\mathcal{A}$. The adversary then sends a pirate decoding box D and two messages m_0, m_1.

4. Challenger first uses Trace_A. In order to do this, it must build a pirate box D_A for scheme $\mathcal{T}_A$ and find two messages $m_{A,0}$ and $m_{A,1}$ for D_A. It does the following:

   (a) Let $T = \lambda \cdot n/\epsilon$. For $i = 1$ to T, it chooses $r^i \leftarrow \mathcal{M}$, computes $\mathsf{ct}_B^i \leftarrow \mathsf{Enc}_B(\mathsf{mpk}_B, r^i)$ and checks if $\mathsf{Test\text{-}Good\text{-}}A(r^i, \mathsf{ct}_B^i) = 1$. If no pair exists, it quits this step. Else, let (r_A, ct_B) be the first such pair.

   (b) Challenger uses (r_A, ct_B) and decoder box D to define D_A and $m_{A,0}, m_{A,1}$. It sets $m_{A,0} = m_0 \oplus r_A$ and $m_{A,1} = m_1 \oplus r_A$. The pirate box D_A has ct_B hardwired, and it takes as input a ciphertext ct and outputs $D((\mathsf{ct}, \mathsf{ct}_B))$. The challenger then computes $z_A \leftarrow \mathsf{Trace}_A^{D_A}(\mathsf{msk}_A, 1^{4/\epsilon}, m_{A,0}, m_{A,1})$. If $z_A \neq \perp$, it outputs z_A and quits.

5. Challenger then uses Trace_B. As before, it must build a pirate box D_B for scheme $\mathcal{T}_B$ and find two messages $m_{B,0}$ and $m_{B,1}$ for D_B. It does the following:

   (a) Let $T = \lambda \cdot n/\epsilon$. For $i = 1$ to T, it chooses $r^i \leftarrow \mathcal{M}$, computes $\mathsf{ct}_A^i \leftarrow \mathsf{Enc}_A(\mathsf{mpk}_A, r^i)$ and checks if $\mathsf{Test\text{-}Good\text{-}}B(r^i, \mathsf{ct}_A^i) = 1$. If no pair exists, it quits this step. Else, let (r_B, ct_A) be the first such pair.

   (b) Challenger uses (r_B, ct_A) and decoder box D to define D_A and $m_{B,0}, m_{B,1}$. It sets $m_{B,0} = m_0 \oplus r_B$ and $m_{B,1} = m_1 \oplus r_B$. The pirate box D_B has ct_A hardwired, and it takes as input a ciphertext ct and outputs $D((\mathsf{ct}_A, \mathsf{ct}))$. The challenger then computes $z_B \leftarrow \mathsf{Trace}_B^{D_B}(\mathsf{msk}_B, 1^{4/\epsilon}, m_{B,0}, m_{B,1})$. If $z_B \neq \perp$, it outputs z_B.

We will define the following events, and the corresponding probabilities. These probabilities are parameterized by the adversary $\mathcal{A}$, polynomial $n(\cdot)$ and non-negligible $\epsilon(\cdot)$, and a function of λ. For simplicity of notations, we will skip the the dependence on $\mathcal{A}$, n and ϵ.

- $\mathsf{Good\text{-}Decoder}$: D distinguishes between encryptions of m_0 and m_1 with advantage ϵ
- Quit_A : No (r_A, ct_B) pair found in Step 4a;
- $\mathsf{Good\text{-}Decoder}_A$: D_A distinguishes between encryptions of $m_{A,0}$ and $m_{A,1}$ with advantage $\epsilon/4$
- $\mathsf{Trace}_A\text{-}\mathsf{Succ}$: $z_A \in S$
- $\mathsf{Trace}_A\text{-}\mathsf{Fail}$: Quit_A or $z_A = \perp$

- Quit_B : No (r_B, ct_A) pair found in Step 5a
- $\mathsf{Good\text{-}Decoder}_B$: D_B distinguishes between encryptions of $m_{B,0}$ and $m_{B,1}$ with advantage $\epsilon/4$
- $\mathsf{Trace}_B\text{-}\mathsf{Succ}$: $z_B \in S$

From the above defined events, it follows that the trace algorithm traces a traitor if one of the following two events occur:

$$\mathcal{E}_1 : \mathsf{Trace}_A\text{-}\mathsf{Succ}$$
$$\text{OR}$$
$$\mathcal{E}_2 : (\mathsf{Trace}_A\text{-}\mathsf{Fail} \wedge \mathsf{Trace}_B\text{-}\mathsf{Succ})$$

As a result, since these events are mutually exclusive, $\Pr[\mathsf{Cor\text{-}Tr}] = \Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2]$. We will now analyse $\Pr[\mathcal{E}_1]$ and $\Pr[\mathcal{E}_2]$ separately.

Theorem 9.1. Assuming $\mathcal{T}_A$ is an f_A-risky secure traitor tracing scheme, for any PPT adversary $\mathcal{A}$, polynomials $p(\cdot), n(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr[\mathcal{E}_1] \geq f_A(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder}] - \text{negl}(\lambda)$.

Proof. First, using the fact that $\mathcal{T}_A$ is an f_A-risky secure traitor tracing scheme, we can relate the probability of event $\mathcal{E}_1$ to the probability of outputting a good pirate box for $\mathcal{T}_A$.

Claim 9.3. Assuming $\mathcal{T}_A$ is an f_A-risky secure traitor tracing scheme, for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr[\mathcal{E}_1] \geq f_A(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder} \wedge \overline{\mathsf{Quit}_A} \wedge \mathsf{Good\text{-}Decoder}_A] - \text{negl}(\lambda)$.

Proof. We will use the PPT adversary $\mathcal{A}$ to build a reduction algorithm $\mathcal{B}$ that interacts with a $\mathcal{T}_A$ challenger and $\mathcal{A}$, and outputs a pirate decoding box D_A and messages $m_{A,0}, m_{A,1}$.

The reduction algorithm $\mathcal{B}$ first receives mpk_A from the challenger. It chooses $(\mathsf{mpk}_B, (\mathsf{sk}_{B,1}, \ldots, \mathsf{sk}_{B,n})) \leftarrow \mathsf{Setup}_B(1^\lambda, 1^n)$ and sends $\mathsf{mpk} = (\mathsf{mpk}_A, \mathsf{mpk}_B)$ to $\mathcal{A}$.

Next, it receives secret key queries from $\mathcal{A}$. For query corresponding to index i, the reduction algorithm sends the same query to challenger. It receives $\mathsf{sk}_{A,i}$, and it sends $\mathsf{sk}_i = (\mathsf{sk}_{A,i}, \mathsf{sk}_{B,i})$ to $\mathcal{A}$.

Finally, after all the queries, the adversary $\mathcal{A}$ sends a pirate decoding box D together with messages m_0, m_1. The reduction algorithm sets $T = \lambda \cdot n/\epsilon$. For $i = 1$ to T, it chooses $r^i \leftarrow \mathcal{M}$, computes $\mathsf{ct}^i_B \leftarrow \mathsf{Enc}_B(\mathsf{mpk}_B, r^i)$ and checks if $\mathsf{Test\text{-}Good\text{-}}A(r^i, \mathsf{ct}^i_B) = 1$. If no pair exists, it outputs an empty decoding box. Else, let (r_A, ct_B) be the first such pair. The reduction algorithm uses (r_A, ct_B) and decoder box D to define D_A and $m_{A,0}, m_{A,1}$. It sets $m_{A,0} = m_0 \oplus r_A$ and $m_{A,1} = m_1 \oplus r_A$. The pirate box D_A has ct_B hardwired, and it takes as input a ciphertext ct and outputs $D((\mathsf{ct}, \mathsf{ct}_B))$. $\mathcal{B}$ sends $D_A, m_{A,0}, m_{A,1}$ to the challenger.

Now, using the security of $\mathcal{T}_A$, it follows that there exists a negligible function $\text{negl}(\cdot)$,

$$\begin{aligned}&\Pr[\mathsf{Trace}_A^{D_A}(\mathsf{msk}_A, 1^{4/\epsilon}, m_{A,0}, m_{A,1}) \in S \wedge \overline{\mathsf{Quit}_A}]\\ &\quad \geq f_A(\lambda, n) \cdot \Pr[\overline{\mathsf{Quit}_A} \wedge \mathsf{Good\text{-}Decoder}_A] - \text{negl}(\lambda).\end{aligned}$$

Since $\Pr[\overline{\mathsf{Quit}_A} \wedge \mathsf{Good\text{-}Decoder}_A] \geq \Pr[\overline{\mathsf{Quit}_A} \wedge \mathsf{Good\text{-}Decoder}_A \wedge \mathsf{Good\text{-}Decoder}]$, it follows that

$$\Pr[\mathcal{E}_1] \geq f_A(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder} \wedge \overline{\mathsf{Quit}_A} \wedge \mathsf{Good\text{-}Decoder}_A] - \text{negl}(\lambda).$$

■

Next, we will show that $\Pr[\mathsf{Good\text{-}Decoder}] - \Pr[\mathsf{Good\text{-}Decoder} \wedge \overline{\mathsf{Quit}_A} \wedge \mathsf{Good\text{-}Decoder}_A]$ is at most a negligible function in λ. First, note that

$$\begin{aligned}\Pr[\mathsf{Good\text{-}Decoder}] &= \Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Quit}_A] + \Pr[\mathsf{Good\text{-}Decoder} \wedge \overline{\mathsf{Quit}_A} \wedge \overline{\mathsf{Good\text{-}Decoder}_A}]\\ &\quad + \Pr[\mathsf{Good\text{-}Decoder} \wedge \overline{\mathsf{Quit}_A} \wedge \mathsf{Good\text{-}Decoder}_A].\end{aligned}$$

Therefore, it suffices to show that $\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Quit}_A]$ and $\Pr[\mathsf{Good\text{-}Decoder} \wedge \overline{\mathsf{Quit}_A} \wedge \overline{\mathsf{Good\text{-}Decoder}_A}]$ are both bounded by negligible funtions.

Lemma 9.3. There exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Quit}_A] \leq \text{negl}(\lambda)$.

Proof. Let $\mathcal{R}_{\mathsf{Enc}_A}$ and $\mathcal{R}_{\mathsf{Enc}_B}$ denote the space of random coins used by Enc_A and Enc_B respectively and for any decoder D, let $\mathsf{Good}_D \subseteq \mathcal{M} \times \mathcal{R}_{\mathsf{Enc}_B}$ denote the set of coins such that for every $(r, r') \in \mathsf{Good}_D$,

$$\Pr\left[D(\mathsf{ct}_A, \mathsf{ct}_B) = b \ : \ \begin{array}{c} b \leftarrow \{0,1\}, \mathsf{ct}_A \leftarrow \mathsf{Enc}_A(\mathsf{mpk}_A, m_b \oplus r) \\ \mathsf{ct}_B = \mathsf{Enc}_B(\mathsf{mpk}_B, r; r') \end{array}\right] \geq \frac{1+\epsilon}{2}.$$

Using a Markov argument, we know that if D is "good decoder", i.e. it distinguishes between encryptions of m_0 and m_1 with advantage at least ϵ, then $|\mathsf{Good}_D|$ is at least $|\mathcal{M}| \cdot |\mathcal{R}_{\mathsf{Enc}_B}| \cdot \epsilon/2$. Concretely, we have that

$$\Pr[(r, r') \in \mathsf{Good}_D \mid \mathsf{Good\text{-}Decoder}] \geq \epsilon/2.$$

Below we show that if $(r, r') \in \mathsf{Good}_D$, then with all but negligible probability, $\mathsf{Test\text{-}Good\text{-}}A$ accepts r and corresponding ciphertext ct_B.

Claim 9.4. There exists a negligible function negl$(\cdot)$ such that for every decoder D, all $\lambda \in \mathbb{N}$, $(r, r') \in \mathsf{Good}_D$,

$$\Pr\left[\mathsf{Test\text{-}Good\text{-}}A(r, \mathsf{Enc}_B(\mathsf{mpk}_B, r; r')) = 1\right] \geq 1 - \text{negl}(\lambda).$$

The proof of above claim follows from Chernoff bounds and is similar to the proof of Lemma C.1. Now, we analyse $\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Quit}_A]$. Note that the event Quit_A occurs if no (r, r') is found after $\lambda \cdot n/\epsilon$ samples, such that $\mathsf{Test\text{-}Good\text{-}}A(r, \mathsf{ct}_B) = 1$ where $\mathsf{ct}_B = \mathsf{Enc}_B(\mathsf{mpk}_B, r; r')$. We argue that this event happens with at most negligible probability. First, we partition the event $\mathsf{Good\text{-}Decoder} \wedge \mathsf{Quit}_A$ into two sub-events:

$$\mathsf{No\text{-}Good\text{-}Pair} : \mathsf{Good\text{-}Decoder} \wedge \text{ None of the } (r, r') \text{ sampled are in } \mathsf{Good}$$
$$\mathsf{Test\text{-}Good\text{-}}A\mathsf{\text{-}Fail} : \mathsf{Good\text{-}Decoder} \wedge \text{Some sample } (r, r') \in \mathsf{Good} \wedge \mathsf{Test\text{-}Good\text{-}}A \text{ rejects } (r, r')$$

From the definition of the events, it follows that $\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Quit}_A] = \Pr[\mathsf{No\text{-}Good\text{-}Pair}] + \Pr[\mathsf{Test\text{-}Good\text{-}}A\mathsf{\text{-}Fail}]$. Let us first analyse $\Pr[\mathsf{No\text{-}Good\text{-}Pair}]$. Recall that

$$\Pr[(r, r') \in \mathsf{Good}_D \mid \mathsf{Good\text{-}Decoder}] \geq \epsilon/2.$$

As a result, the probability of not finding $(r, r') \in \mathsf{Good}$ after $T = \lambda/\epsilon$ samples is at most $(1 - \epsilon/2)^T \leq 2^{-O(\lambda)}$, which is negligible in λ. Next, from Claim 9.4, it follows that there exists a negligible function negl$_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[\mathsf{Test\text{-}Good\text{-}}A\mathsf{\text{-}Fail}] \leq \text{negl}_2(\lambda)$. Thus, we get that $\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Quit}_A] \leq \text{negl}(\lambda)$. ∎

Lemma 9.4. There exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\Pr[\mathsf{Good\text{-}Decoder} \wedge \overline{\mathsf{Quit}_A} \wedge \overline{\mathsf{Good\text{-}Decoder}_A}] \leq \text{negl}(\lambda).$$

Proof. Let $\mathsf{Bad}_D \subseteq \mathcal{M} \times \mathcal{R}_{\mathsf{Enc}_B}$ denote the set of coins such that for every $(r, r') \in \mathsf{Bad}_D$, decoder $D_A = D(\cdot, \mathsf{Enc}_B(\mathsf{mpk}_B, r; r'))$ is a "bad" decoder for system A, i.e. D_A is not an $(\epsilon/4)$-Dist decoder for $m_{A,0}, m_{A,1}$. Using Chernoff bounds, we get that for every decoder D, $(r, r') \in \mathsf{Bad}_D$,

$$\Pr\left[\mathsf{Test\text{-}Good\text{-}}A(r, \mathsf{Enc}_B(\mathsf{mpk}_B, r; r')) = 1\right] \leq 2^{-O(\lambda)}.$$

Now note that while tracing using Trace_A, routine $\mathsf{Test\text{-}Good\text{-}}A$ is independently run $T = \lambda \cdot n/\epsilon$ times. Using a union bound, we get that the probability $\mathsf{Test\text{-}Good\text{-}}A$ accepts a pair $(r, r') \in \mathsf{Bad}_D$, in any of those T tries, is at most $T \cdot 2^{-O(\lambda)} = \text{negl}(\lambda)$. Thus, we can conclude that $\Pr[\overline{\mathsf{Quit}_A} \wedge \overline{\mathsf{Good\text{-}Decoder}_A}] \leq \text{negl}(\lambda)$. This completes the proof. ∎

From the above two lemmas, we can conclude that there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[\mathcal{E}_1] \geq f_A(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder}] - \mathrm{negl}(\lambda)$. Similarly, we show a lower bound on $\Pr[\mathcal{E}_2]$ next.

Theorem 9.2. Assuming $\mathcal{T}_B$ is an f_B-risky secure traitor tracing scheme, for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) \geq 1/p(\lambda)$,

$$\Pr[\mathcal{E}_2] \geq f_B(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\text{-}\mathsf{Fail}].$$

Proof. The proof of this theorem will follow a similar structure as the proof of Theorem 9.1. First, using the fact that $\mathcal{T}_B$ is an f_B-risky secure traitor tracing scheme, we can relate the probability of event $\mathcal{E}_2$ to the probability of outputting a good pirate box for $\mathcal{T}_B$.

Claim 9.5. Assuming $\mathcal{T}_B$ is an f_B-risky secure traitor tracing scheme, for every PPT adversary $\mathcal{A}$, polynomial $n(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[\mathcal{E}_2] \geq f_B(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\text{-}\mathsf{Fail} \wedge \overline{\mathsf{Quit}_B} \wedge \mathsf{Good\text{-}Decoder}_B] - \mathrm{negl}(\lambda)$.

Proof. We will use the PPT adversary $\mathcal{A}$ to build a reduction algorithm $\mathcal{B}$ that interacts with a $\mathcal{T}_B$ challenger and $\mathcal{A}$, and outputs a pirate decoding box D_B and messages $m_{B,0}, m_{B,1}$.

The reduction algorithm $\mathcal{B}$ first receives mpk_B from the challenger. It chooses $(\mathsf{mpk}_A, (\mathsf{sk}_{A,1}, \ldots, \mathsf{sk}_{A,n})) \leftarrow \mathsf{Setup}_A(1^\lambda, 1^n)$ and sends $\mathsf{mpk} = (\mathsf{mpk}_A, \mathsf{mpk}_B)$ to $\mathcal{A}$.

Next, it receives secret key queries from $\mathcal{A}$. For query corresponding to index i, the reduction algorithm sends the same query to challenger. It receives $\mathsf{sk}_{\mathcal{B},i}$, and it sends $\mathsf{sk}_i = (\mathsf{sk}_{A,i}, \mathsf{sk}_{B,i})$ to $\mathcal{A}$. Let S denote the set of keys queried.

Finally, after all the queries, the adversary $\mathcal{A}$ sends a pirate decoding box D together with messages m_0, m_1. The reduction algorithm first 'simulates' the tracing of traitors using Trace_A. It performs Trace attempt using Trace_A (that is, Step 4). If it successfully traces an index $i \in S$, then the reduction algorithm quits (that is, it outputs an empty tracing box).

Else, the reduction algorithm sets $T = \lambda \cdot n/\epsilon$. For $i = 1$ to T, it chooses $r^i \leftarrow \mathcal{M}$, computes $\mathsf{ct}_A^i \leftarrow \mathsf{Enc}_A(\mathsf{mpk}_A, r^i)$ and checks if $\mathsf{Test\text{-}Good\text{-}}B(r^i, \mathsf{ct}_A) = 1$. If no pair exists, it outputs an empty decoding box. Else, let (r_B, ct_A) be the first such pair. The reduction algorithm uses (r_B, ct_A) and decoder box D to define D_B and $m_{B,0}, m_{B,1}$. It sets $m_{B,0} = m_0 \oplus r_B$ and $m_{B,1} = m_1 \oplus r_B$. The pirate box D_B has ct_A hardwired, and it takes as input a ciphertext ct and outputs $D((\mathsf{ct}_A, \mathsf{ct}))$. $\mathcal{B}$ sends $D_B, m_{B,0}, m_{B,1}$ to the challenger.

Now, from security of $\mathcal{T}_B$, it follows that

$$\begin{aligned}\Pr[\mathcal{E}_2] &= \Pr[\mathsf{Trace}_B^{D_B}(\mathsf{msk}_B, 1^{4/\epsilon}, m_{B,0}, m_{B,1}) \in S \wedge \mathsf{Trace}_A\text{-}\mathsf{Fail} \wedge \overline{\mathsf{Quit}_B}] \\ &\geq f_B(\lambda, n) \cdot \Pr[\mathsf{Trace}_A\text{-}\mathsf{Fail} \wedge \overline{\mathsf{Quit}_B} \wedge \mathsf{Good\text{-}Decoder}_B] - \mathrm{negl}(\lambda) \\ &\geq f_B(\lambda, n) \cdot \Pr[\mathsf{Trace}_A\text{-}\mathsf{Fail} \wedge \overline{\mathsf{Quit}_B} \wedge \mathsf{Good\text{-}Decoder}_B \wedge \mathsf{Good\text{-}Decoder}] - \mathrm{negl}(\lambda).\end{aligned}$$

This concludes the proof.

■

Next, in order to show that $\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\text{-}\mathsf{Fail} \wedge \overline{\mathsf{Quit}_B} \wedge \mathsf{Good\text{-}Decoder}_B] \geq \Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\text{-}\mathsf{Fail}] - \mathrm{negl}(\lambda)$, it suffices to show that $\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\text{-}\mathsf{Fail} \wedge \mathsf{Quit}_B]$ and $\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\text{-}\mathsf{Fail} \wedge \overline{\mathsf{Quit}_B} \wedge \overline{\mathsf{Good\text{-}Decoder}_B}]$ are both bounded by some negligible funtions.

Lemma 9.5. There exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\text{-}\mathsf{Fail} \wedge \mathsf{Quit}_B] \leq \mathrm{negl}(\lambda)$.

Proof. The proof of this lemma is very similar to the proof of Lemma 9.3, so we will briefly list the modifications required.

The set of "good" coins corresponding to a decoder D will now be defined as $\mathsf{Good}_D \subseteq \mathcal{M} \times \mathcal{R}_{\mathsf{Enc}_A}$ such that for every $(r, r') \in \mathsf{Good}_D$,

$$\Pr\left[D(\mathsf{ct}_A, \mathsf{ct}_B) = b \ : \ \begin{array}{c} b \leftarrow \{0,1\}, \mathsf{ct}_B \leftarrow \mathsf{Enc}_B(\mathsf{mpk}_B, m_b \oplus r) \\ \mathsf{ct}_A = \mathsf{Enc}_A(\mathsf{mpk}_A, r; r') \end{array}\right] \geq \frac{1+\epsilon}{2}.$$

Next, using a Markov argument, we will argue that fraction of coins in Good_D for a good decoder is at least $\epsilon/2$. For completing this argument, the following observation will be important.

Observation 9.1. For any $m \in \mathcal{M}$, the following distributions are identical:

$$\{(m \oplus r, r) : r \leftarrow \mathcal{M}\} \equiv \{(r, m \oplus r) : r \leftarrow \mathcal{M}\}$$

The rest of the proof will be identical in which we will first argue that $\mathsf{Test\text{-}Good\text{-}}B$ accepts all $(r, r') \in \mathsf{Good}_D$ with all but negligible probability. Next, we will divide the event $\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\mathsf{\text{-}Fail} \wedge \mathsf{Quit}_B$ into two sub-events as before and argue that they both occur with at most negligible probability.

■

Lemma 9.6. There exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\mathsf{\text{-}Fail} \wedge \overline{\mathsf{Quit}_B} \wedge \overline{\mathsf{Good\text{-}Decoder}_B}] \leq \mathrm{negl}(\lambda).$$

The proof of this lemma is identical to that of Lemma 9.4. Combining the above lemmas, we get that there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[\mathcal{E}_2] \geq f_B(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\mathsf{\text{-}Fail}] - \mathrm{negl}(\lambda)$.

■

Combining Theorem 9.1 and Theorem 9.2, we can compute the 'riskyness' of our traitor tracing scheme. Concretely, we get that

$$\begin{aligned}
&\Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2] \\
&\quad \geq \Pr[\mathcal{E}_1] + f_B(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\mathsf{\text{-}Fail}] - \mathrm{negl}_1(\lambda) \\
&\quad = \Pr[\mathcal{E}_1] + f_B(\lambda, n) \cdot (\Pr[\mathsf{Good\text{-}Decoder}] - \Pr[\mathcal{E}_1] - \Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda)) - \mathrm{negl}_1(\lambda) \\
&\quad \geq \Pr[\mathcal{E}_1] \cdot (1 - f_B(\lambda, n)) + f_B(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder}] - \mathrm{negl}_2(\lambda) \\
&\quad \geq f_A(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder}] \cdot (1 - f_B(\lambda, n)) + f_B(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder}] - \mathrm{negl}_3(\lambda) \\
&\quad \geq (f_A(\lambda, n) + f_B(\lambda, n) - f_A(\lambda, n) \cdot f_B(\lambda, n)) \cdot \Pr[\mathsf{Good\text{-}Decoder}] - \mathrm{negl}_3(\lambda)
\end{aligned}$$

This concludes the proof.

■

References

[BF99] Dan Boneh and Matthew K. Franklin. An efficient public key traitor tracing scheme. In *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, pages 338–353, 1999.

[BGDM+10] Jean-Luc Beuchat, Jorge E González-Díaz, Shigeo Mitsunari, Eiji Okamoto, Francisco Rodríguez-Henríquez, and Tadanori Teruya. High-speed software implementation of the optimal ate pairing over barreto–naehrig curves. In *International Conference on Pairing-Based Cryptography*, pages 21–39. Springer, 2010.

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *CRYPTO*, pages 1–18, 2001.

[BN08] Dan Boneh and Moni Naor. Traitor tracing with constant size ciphertext. In *Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, Virginia, USA, October 27-31, 2008*, pages 501–510, 2008.

[BP08] Olivier Billet and Duong Hieu Phan. Efficient traitor tracing from collusion secure codes. In *Information Theoretic Security, Third International Conference, ICITS 2008, Calgary, Canada, August 10-13, 2008, Proceedings*, pages 171–182, 2008.

[BSW06] Dan Boneh, Amit Sahai, and Brent Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, pages 573–592, 2006.

[BW06] Dan Boneh and Brent Waters. A fully collusion resistant broadcast, trace, and revoke system. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 211–220. ACM, 2006.

[BWY11] Mihir Bellare, Brent Waters, and Scott Yilek. Identity-based encryption secure against selective opening attack. In *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings*, pages 235–252, 2011.

[BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, pages 480–499, 2014.

[CFN94] Benny Chor, Amos Fiat, and Moni Naor. Tracing traitors. In *CRYPTO*, pages 257–270, 1994.

[CFNP00] Benny Chor, Amos Fiat, Moni Naor, and Benny Pinkas. Tracing traitors. *IEEE Transactions on Information Theory*, 46(3):893–910, 2000.

[CPP05] Hervé Chabanne, Duong Hieu Phan, and David Pointcheval. Public traceability in traitor tracing schemes. In *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, pages 542–558, 2005.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, pages 265–284, 2006.

[DNR$^+$09] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil Vadhan. On the complexity of differentially private data release: Efficient algorithms and hardness results. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 381–390, New York, NY, USA, 2009. ACM.

[Fre10] David Mandell Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, pages 44–61, 2010.

[GGH$^+$13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, 2013.

[GKSW10] Sanjam Garg, Abishek Kumarasubramanian, Amit Sahai, and Brent Waters. Building efficient fully collusion-resilient traitor tracing and revocation schemes. In *Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010*, pages 121–130, 2010.

[GKW18] Rishab Goyal, Venkata Koppula, and Brent Waters. Collusion resistant traitor tracing from learning with errors. 2018.

[KMUW17] Lucas Kowalczyk, Tal Malkin, Jonathan Ullman, and Daniel Wichs. Hardness of non-interactive differential privacy from one-way functions. Cryptology ePrint Archive, Report 2017/1107, 2017. https://eprint.iacr.org/2017/1107.

[KMUZ16] Lucas Kowalczyk, Tal Malkin, Jonathan Ullman, and Mark Zhandry. Strong hardness of privacy from weak traitor tracing. In *Proceedings of TCC 2016-B*, 2016.

[KP10] Aggelos Kiayias and Serdar Pehlivanoglu. *Encryption for Digital Content*, volume 52 of *Advances in Information Security*. Springer, 2010.

[KY02] Aggelos Kiayias and Moti Yung. Traitor tracing with constant transmission rate. In *Advances in CryptologyEUROCRYPT 2002*, pages 450–465. Springer, 2002.

[KY03] Aggelos Kiayias and Moti Yung. Breaking and repairing asymmetric public-key traitor tracing. In *Digital Rights Management: ACM CCS-9 Workshop, DRM 2002, Washington, DC, USA, November 18, 2002, Revised Papers*, 2003.

[LW10] Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, pages 455–479, 2010.

[NP98] Moni Naor and Benny Pinkas. Threshold traitor tracing. In *Advances in Cryptology-CRYPTO'98*, pages 502–517. Springer, 1998.

[NP00] Moni Naor and Benny Pinkas. Efficient trace and revoke schemes. In *Financial Cryptography, 4th International Conference, FC 2000 Anguilla, British West Indies, February 20-24, 2000, Proceedings*, pages 1–20, 2000.

[NWZ16] Ryo Nishimaki, Daniel Wichs, and Mark Zhandry. Anonymous traitor tracing: How to embed arbitrary information in a key. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, pages 388–419, 2016.

[Pfi96] Birgit Pfitzmann. Trials of traced traitors. In *Information Hiding*, pages 49–64. Springer, 1996.

[PW97] Birgit Pfitzmann and Michael Waidner. Asymmetric fingerprinting for larger collusions. In *Proceedings of the 4th ACM conference on Computer and communications security*, pages 151–160. ACM, 1997.

[rel] Relic toolkit. https://github.com/relic-toolkit/relic.

[Sir06] Thomas Sirvent. Traitor tracing scheme with constant ciphertext rate against powerful pirates. Cryptology ePrint Archive, Report 2006/383, 2006. http://eprint.iacr.org/2006/383.

[Ull13] Jonathan Ullman. Answering $n_{\{2+o(1)\}}$ counting queries with differential privacy is hard. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 361–370, 2013.

[WHI01] Yuji Watanabe, Goichiro Hanaoka, and Hideki Imai. Efficient asymmetric public-key traitor tracing without trusted agents. *Topics in CryptologyCT-RSA 2001*, pages 392–407, 2001.

A Missing Proofs from Section 4

A.1 Proof of Theorem 4.1

Proof. We prove this theorem via a series of hybrid experiments. First, for all $\ell \in \mathbb{N}$ consider the following helper function $\mathsf{Zip} : \{0,1\}^\ell \times \{0,1\}^\ell \times \{0,\dots,\ell\} \to \{0,1\}^\ell$:

Definition A.1.

$$\mathsf{Zip}(\mathbf{y}_0, \mathbf{y}_1, j)_i = \begin{cases} y_{0,i} \wedge y_{1,i} & \text{if } i \le j \\ y_{0,i} & \text{otherwise} \end{cases}$$

Note that for any $\mathbf{y}_0, \mathbf{y}_1$, $\mathsf{Zip}(\mathbf{y}_0, \mathbf{y}_1, 0) = \mathbf{y}_0$. Next, we prove the following lemma about this function:

Lemma A.1. Let $\ell \in \mathbb{N}$. For any vectors $\mathbf{x}, \mathbf{y}_0, \mathbf{y}_1 \in \{0,1\}^\ell$ and $j \in \{0,\dots,\ell-1\}$, if $f(\mathbf{x},\mathbf{y}_0) = f(\mathbf{x},\mathbf{y}_1)$ then $f(\mathbf{x},\mathbf{w}_0) = f(\mathbf{x},\mathbf{w}_1)$ where $\mathbf{w}_0 = \mathsf{Zip}(\mathbf{y}_0,\mathbf{y}_1,j)$ and $\mathbf{w}_1 = \mathsf{Zip}(\mathbf{y}_0,\mathbf{y}_1,j+1)$.

Proof. Note that $w_{0,i} = w_{1,i}$ for all $i \ne j+1$. Thus, if $x_{j+1} = 1$ or if $w_{0,j+1} = w_{1,j+1}$ then $f(\mathbf{x},\mathbf{w}_0) = f(\mathbf{x},\mathbf{w}_1)$.

We then consider the case where $w_{0,j+1} \ne w_{1,j+1}$ and $x_{j+1} = 0$. By definition, $w_{0,j+1} = y_{0,j+1}$ and $w_{1,j+1} = y_{0,j+1} \wedge y_{1,j+1}$. Hence, $y_{0,j+1} = 1$ and $y_{1,j+1} = 0$. Since $x_{j+1} = 0$ (and $y_{1,j+1} = 0$), $f(\mathbf{x},\mathbf{y}_1) = 0$ and $f(\mathbf{x},\mathbf{w}_1) = 0$.

Also, note that since $f(\mathbf{x},\mathbf{y}_0) = f(\mathbf{x},\mathbf{y}_1)$, it follows that $f(\mathbf{x},\mathbf{y}_0) = 0$, and therefore there exists an index j^* such that $x_{j^*} = y_{0,j^*} = 0$. If $j^* > j+1$, then $w_{0,j^*} = y_{0,j^*} = 0$. If $j^* < j+1$, then $w_{0,j^*} = y_{0,j^*} \wedge y_{1,j^*} = 0$. Since $y_{0,j^*} = 1$, $j^* \ne j+1$, and therefore there exists an index j^* such that $w_{j^*} = x_{j^*} = 0$. This implies that $f(\mathbf{x},\mathbf{w}_0) = 0$. This concludes our proof.

■

Now we define our hybrid experiments using Zip. Let $\lambda \in \mathbb{N}$, $\ell(\cdot)$ be a polynomial. For any stateful PPT adversary $\mathcal{A}$ and $j \in \{0,\dots,\ell(\lambda)\}$, define $\mathsf{Hyb}^{(j)}_{\mathsf{mBME},\ell(\lambda),\mathcal{A}}(1^\lambda)$ in Figure 13.

Experiment $\mathsf{Hyb}^{(j)}_{\mathsf{mBME},\ell(\lambda),\mathcal{A}}(1^\lambda)$

- $(\mathbf{y}_0, \mathbf{y}_1) \leftarrow \mathcal{A}(1^\lambda)$.
- $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell(\lambda)})$
- $(m_0, m_1) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{pk})$
- $b \leftarrow \{0,1\}$, $\mathsf{ct}_b \leftarrow \mathsf{Enc\text{-}SK}(\mathsf{msk}, m_b, \mathbf{z})$ where $\mathbf{z} = \mathsf{Zip}(\mathbf{y}_b, \mathbf{y}_{1-b}, j)$
- $b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{ct}_b)$
- Output 1 if $b = b'$, and 0 otherwise.

Adversarial Restrictions: For all queries $\mathbf{x}$ made by $\mathcal{A}$ to $\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}$ the following conditions must hold:

- If $m_0 = m_1$, then $f(\mathbf{x},\mathbf{y}_0) = f(\mathbf{x},\mathbf{y}_1)$.
- If $m_0 \ne m_1$, then $f(\mathbf{x},\mathbf{y}_0) = f(\mathbf{x},\mathbf{y}_1) = 0$.

Figure 13: Ciphertext Hybrid (j)

For notational purposes, we will consider the parameters $\mathcal{A}$, mBME, $\ell(\cdot)$, and λ implicitly where clear, and write $\mathsf{Hyb}^{(j)}_{\mathsf{mBME},\ell(\lambda),\mathcal{A}}(1^\lambda)$ as Hyb_j and $\ell(\lambda)$ as ℓ.

Note that $\mathsf{Expt\text{-}ct\text{-}ind}^{\mathcal{A}} = \mathsf{Hyb}_0^{\mathcal{A}}$; this is because $\mathsf{Zip}(\mathbf{y}_b, \mathbf{y}_{1-b}, 0) = \mathbf{y}_b$ and therefore, $\Pr[1 \leftarrow \mathsf{Expt\text{-}ct\text{-}ind}^{\mathcal{A}}] = \Pr[1 \leftarrow \mathsf{Hyb}_0^{\mathcal{A}}]$.

Claim A.1. Let $j \in \{0, \dots, \ell(\lambda) - 1\}$, and $\mathcal{A}$ be any stateful PPT adversary. If mBME satisfies 1-attribute hiding, then $\Pr[1 \leftarrow \mathsf{Hyb}_{j+1}^{\mathcal{A}}] - \Pr[1 \leftarrow \mathsf{Hyb}_j^{\mathcal{A}}] \leq \text{negl}(\lambda)$.

Proof. Suppose there exists a PPT adversary such that $p_{\mathcal{A},j+1} = \Pr[1 \leftarrow \mathsf{Hyb}_{j+1}^{\mathcal{A}}]$, $p_{\mathcal{A},j} = \Pr[1 \leftarrow \mathsf{Hyb}_j^{\mathcal{A}}]$ and $p_{\mathcal{A},j+1} - p_{\mathcal{A},j} = \epsilon$. We construct a stateful PPT adversary $\mathcal{B}$ such that $\Pr[1 \leftarrow \mathsf{Expt\text{-}1\text{-}attr\text{-}ct\text{-}ind}^{\mathcal{B}}] = \epsilon$ using $\mathcal{A}$. The reduction algorithm $\mathcal{B}$ first receives $(\mathbf{y}_0, \mathbf{y}_1)$ from the adversary $\mathcal{A}$. It chooses $b \leftarrow \{0, 1\}$, sets $\mathbf{w}_0 = \mathsf{Zip}(\mathbf{y}_b, \mathbf{y}_{1-b}, j + 1)$ and $\mathbf{w}_1 = \mathsf{Zip}(\mathbf{y}_b, \mathbf{y}_{1-b}, j)$ and sends $(\mathbf{w}_0, \mathbf{w}_1)$ to the challenger. The challenger sends a public key pk, which the reduction algorithm forwards to the adversary.

Next, the adversary is allowed to make pre-challenge key/ciphertext queries. All queries are forwarded to the challenger. The challenger's response is then forwarded to the adversary $\mathcal{A}$. The adversary then sends its challenge messages m_0, m_1. The reduction algorithm sends message m_b to the challenger, and receives a ciphertext ct^*, which it forwards to the adversary. The post-challenge key/ciphertext queries are handled similar to the pre-challenge queries. Finally, the adversary sends its guess b'. If $b = b'$, then the reduction algorithm guesses that it received an encryption of m_b for $\mathbf{w}_1$, else it guesses that ct^* is an encryption of m_b for $\mathbf{w}_0$.

Analysis : First, note that if $\mathcal{A}$ is an admissible adversary for the hybrid experiments, then $\mathcal{B}$ is an admissible adversary for the 1-attribute hiding experiment. This is because for every key query $\mathbf{x}$ sent by $\mathcal{A}$, $f(\mathbf{x}, \mathbf{y}_0) = f(\mathbf{x}, \mathbf{y}_1)$. As a result, using Lemma A.1, it follows that $f(\mathbf{x}, \mathbf{w}_0) = f(\mathbf{x}, \mathbf{w_1})$.

Let us now analyse the advantage of $\mathcal{B}$. Note that if the challenger encrypts for $\mathbf{w}_0$, then $\mathcal{B}$ perfectly simulates Hyb_{j+1} for $\mathcal{A}$, and if the challenger encrypts for $\mathbf{w}_1$, then $\mathcal{B}$ perfectly simulates Hyb_j for $\mathcal{A}$.

$$\begin{aligned}\mathsf{Adv}_{\mathcal{B}} &= 1/2(\Pr[\mathcal{B} \text{ guesses } \mathbf{w_0} : \text{Challenger encrypts for } \mathbf{w}_0] + \Pr[\mathcal{B} \text{ guesses } \mathbf{w_0} : \text{Challenger encrypts for } \mathbf{w}_1]) - 1/2 \\ &= 1/2(p_{\mathcal{A},j+1} + (1 - p_{\mathcal{A},j})) - 1/2 = \epsilon/2.\end{aligned}$$

■

Claim A.2. If mBME satisfies ciphertext indistinguishability under chosen attributes, then for any PPT adversary $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[1 \leftarrow \mathsf{Hyb}_\ell^{\mathcal{A}}] - \Pr[1 \leftarrow \mathsf{Hyb}_\ell^{\mathcal{A}}] \leq \text{negl}(\lambda)$.

Proof. Suppose that there exists a PPT adversary $\mathcal{A}$ and a non-negligible function ϵ such that $\Pr[1 \leftarrow \mathsf{Hyb}_\ell^{\mathcal{A}}] - \Pr[1 \leftarrow \mathsf{Hyb}_\ell^{\mathcal{A}}] = \epsilon$. We construct a stateful PPT adversary $\mathcal{B}$ such that $\Pr[1 \leftarrow \mathsf{Expt\text{-}IND\text{-}CA}^{\mathcal{B}}] = 1/2 + \epsilon$. The reduction algorithm $\mathcal{B}$ first receives $(\mathbf{y_0}, \mathbf{y_1})$ from the adversary. It computes $\mathbf{w} = \mathsf{Zip}(\mathbf{y}_0, \mathbf{y}_1, \ell)$ and sends $\mathbf{w}$ to the challenger. The challenger sends pk to $\mathcal{B}$, which it forwards to the adversary.

Next, the adversary is allowed to make pre-challenge key/ciphertext queries. The reduction algorithm forwards these queries to the challenger, and then forwards the challenger's response to $\mathcal{A}$. The adversary then sends challenge messages m_0, m_1. The reduction algorithm sends m_0, m_1 as challenge messages, and it receives challenge ciphertext ct^*, which is forwarded to $\mathcal{A}$. The post-challenge queries are handled similar to the pre-challenge ones. Finally, the adversary submits its guess b', which the reduction algorithm forwards to the challenger.

Analysis We now show that $\mathcal{B}$ is a valid adversary for $\mathsf{Expt\text{-}IND\text{-}CA}$. If $m_0 \neq m_1$, then every key query $\mathbf{x}$ made by $\mathcal{A}$ must satisfy $f(\mathbf{x}, \mathbf{y}_0) = f(\mathbf{x}, \mathbf{y}_1) = 0$. This implies that $f(\mathbf{x}, \mathbf{w}) = 0$. Hence all key queries made by $\mathcal{B}$ are admissible. Also note that since $\mathsf{Zip}(\mathbf{y}_0, \mathbf{y}_1, \ell) = \mathsf{Zip}(\mathbf{y}_1, \mathbf{y}_0, \ell) = \mathbf{w}$, $\mathcal{B}$ perfectly simulates Hyb_ℓ for $\mathcal{A}$.

We will now compute $\mathcal{B}$'s advantage. Note that $\mathcal{B}$ wins the ciphertext indistinguishability game if and only if $\mathcal{A}$ wins in Hyb_ℓ. Therefore, the advantage of $\mathcal{B}$ is identical to the advantage of $\mathcal{A}$. This concludes our proof.

■

Proof of Theorem 4.1 Finally, under our assumptions that mBME satisfies 1-attribute hiding and ciphertext indistinguishability under chosen attributes, both $\Pr[1 \leftarrow \mathsf{Expt\text{-}1\text{-}attr\text{-}ct\text{-}ind}^{\mathcal{A}}] \leq \text{negl}(\lambda)$ and $\Pr[1 \leftarrow \mathsf{Expt\text{-}IND\text{-}CA}^{\mathcal{A}}] \leq \text{negl}(\lambda)$ for all stateful PPT adversaries $\mathcal{A}$ by our two claims. Thus,

$$\begin{aligned}
\Pr[1 \leftarrow \mathsf{Expt\text{-}ct\text{-}ind}^{\mathcal{A}}] =& \Pr[1 \leftarrow \mathsf{Hyb}_0^{\mathcal{A}}] \\
\leq& \Pr[1 \leftarrow \mathsf{Hyb}_0^{\mathcal{A}}] + \sum_{i=1}^{\ell} \left(\Pr[1 \leftarrow \mathsf{Hyb}_i^{\mathcal{A}}] - \Pr[1 \leftarrow \mathsf{Hyb}_{i-1}^{\mathcal{A}}] \right) \\
\leq& \Pr[1 \leftarrow \mathsf{Hyb}_\ell^1] + \ell \cdot \text{negl}(\lambda) \\
\leq& 1/2 + \text{negl}(\lambda)
\end{aligned}$$

Thus, mBME satisfies ciphertext hiding. ■

A.2 Proof of Theorem 4.2

Proof. We reduce between these notions of security via a series of hybrids, following a similar approach as the proof for Theorem 4.1. First, recall the definition of the helper function Zip (Definition A.1). We will present an analogue of Lemma A.1 for the key-hiding proof.

Lemma A.2. Let $\ell \in \mathbb{N}$. For any vectors $\mathbf{x}_0, \mathbf{x}_1, \mathbf{y} \in \{0,1\}^\ell$ and $j \in \{0, \ldots, \ell-1\}$, if $f(\mathbf{x}_0, \mathbf{y}) = f(\mathbf{x}_1, \mathbf{y})$ then $f(\mathbf{w}_0, \mathbf{y}) = f(\mathbf{w}_1, \mathbf{y})$ where $\mathbf{w}_0 = \mathsf{Zip}(\mathbf{x}_0, \mathbf{x}_1, j)$ and $\mathbf{w}_1 = \mathsf{Zip}(\mathbf{x}_0, \mathbf{x}_1, j+1)$.

Let $\lambda \in \mathbb{N}$, $\ell(\cdot)$ be a polynomial. For any stateful PPT adversary $\mathcal{A}$, any $b \in \{0,1\}$ and $j \in \{0, \ldots, \ell(\lambda)\}$, define $\mathsf{Hyb}^{(j)}_{\mathsf{mBME},\ell(\lambda),\mathcal{A}}(1^\lambda)$ in Figure 14. For notational purposes, we will consider the parameters $\mathcal{A}$, mBME, $\ell(\cdot)$, and λ implicitly where clear, and write $\mathsf{Hyb}^{(j)}_{\mathsf{mBME},\ell(\lambda),\mathcal{A}}(1^\lambda)$ as Hyb_j and $\ell(\lambda)$ as ℓ.

Note that $\mathsf{Expt\text{-}key\text{-}ind}^{\mathcal{A}} = \mathsf{Hyb}_0^{\mathcal{A}}$, and thus, $\Pr[1 \leftarrow \mathsf{Expt\text{-}key\text{-}ind}^{\mathcal{A}}] = \Pr[1 \leftarrow \mathsf{Hyb}_0^{\mathcal{A}}]$.

Experiment $\mathsf{Hyb}^{(j)}_{\mathsf{mBME},\ell(\lambda),\mathcal{A}}(1^\lambda)$

- $(\mathbf{x}_0, \mathbf{x}_1) \leftarrow \mathcal{A}(1^\lambda)$.
- $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell(\lambda)})$.
- $b \leftarrow \{0,1\}$, $\mathsf{sk}_b \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathbf{z}_b)$ where $\mathbf{z}_b = \mathsf{Zip}(\mathbf{x}_b, \mathbf{x}_{1-b}, j)$.
- $b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{sk}}_{\mathsf{msk}}, \mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}}(\mathsf{pk}, \mathsf{ct}_b)$
- Output 1 if $b = b'$, and 0 otherwise.

Adversarial Restrictions: For all queries $(m, \mathbf{y})$ made by $\mathcal{A}$ to $\mathcal{O}^{\mathsf{ct}}_{\mathsf{msk}}$ the following equality must hold: $f(\mathbf{x_0}, \mathbf{y}) = f(\mathbf{x_1}, \mathbf{y})$.

Figure 14: Key Hybrid (j)

Lemma A.3. Assuming the mBME scheme is selective 1-attribute key hiding, for any PPT adversary $\mathcal{A}$ and index $j \in \{0, 1, \ldots, \ell-1\}$, there exists a negligible function $\text{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[1 \leftarrow \mathsf{Hyb}_{j+1}^{\mathcal{A}}] - \Pr[1 \leftarrow \mathsf{Hyb}_j^{\mathcal{A}}] \leq \text{negl}(\lambda)$.

Suppose there exists a PPT adversary $\mathcal{A}$ and a non-negligible function ϵ such that $\Pr[1 \leftarrow \mathsf{Hyb}_{j+1}^{\mathcal{A}}] - \Pr[1 \leftarrow \mathsf{Hyb}_j^{\mathcal{A}}] = \epsilon$. We construct a PPT adversary $\mathcal{B}$ such that $\Pr[1 \leftarrow \mathsf{Expt\text{-}1\text{-}attr\text{-}key\text{-}ind}^{\mathcal{B}}] = 1/2 + \epsilon$. The reduction algorithm $\mathcal{B}$ first receives the challenge vectors $\mathbf{x}_0, \mathbf{x}_1$ from the adversary $\mathcal{A}$. It then chooses a bit $b \leftarrow \{0,1\}$, computes $\mathbf{w}_0 = \mathsf{Zip}(\mathbf{x}_b, \mathbf{x}_{1-b}, j+1)$ and $\mathbf{w}_1 = \mathsf{Zip}(\mathbf{x}_b, \mathbf{x}_{1-b}, j)$, and sends $\mathbf{w}_0, \mathbf{w}_1$ to the challenger. The challenger sends pk and sk^* to $\mathcal{B}$, which it forwards to the adversary. Next, the adversary is allowed to make key/ciphertext queries to the reduction algorithm. The reduction algorithm forwards the queries to the challenger, and then forwards the challenger's response to $\mathcal{A}$. Finally, $\mathcal{A}$ sends its guess b'. If $b = b'$, the reduction algorithm guesses that the key corresponds to $\mathbf{w}_0$, else it guesses that it corresponds to $\mathbf{w}_1$.

Analysis We now show that $\mathcal{B}$ is a valid adversary for $\mathsf{Expt\text{-}1\text{-}attr\text{-}key\text{-}ind}$. First, note that $\mathbf{w}_0, \mathbf{w}_1$ can differ at only one position (index $j+1$). Next, using Lemma A.2, it follows that $f(\mathbf{w}_0, \mathbf{y}) = f(\mathbf{w}_1, \mathbf{y})$ if $f(\mathbf{x}_0, \mathbf{y}) = f(\mathbf{x}_1, \mathbf{y})$. Finally, as in the ciphertext hiding proof, we can show that $\Pr[1 \leftarrow \mathsf{Hyb}^{\mathcal{A}}_{j+1}] - \Pr[1 \leftarrow \mathsf{Hyb}^{\mathcal{A}}_{j}] = \epsilon$. Thus, mBME satisfies key hiding.

■

B Preliminaries (Cont'd)

B.1 Bilinear Groups and Assumptions

In this work, we will use composite order bilinear groups for our main construction. Let $\mathbb{G}, \mathbb{G}_T$ be two groups of order N. A bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is an efficiently computable function mapping two group elements of $\mathbb{G}$ to a group element in $\mathbb{G}_T$ and satisfying the following properties:

- Bilinearity : $\forall g \in \mathbb{G}, a, b \in \mathbb{Z}_N, e(g^a, g^b) = e(g, g)^{ab}$
- Non-Degeneracy : $e(g, g) \neq 1_{G_T}$ for $g \neq 1_G$, where 1_G and 1_{G_T} are the identity elements of $\mathbb{G}$ and $\mathbb{G}_T$ respectively.

We will now present different assumptions on composite order bilinear groups. First, we will present the a generalization of the subgroup decision assumption introduced by Bellare, Waters and Yilek [BWY11]. Essentially they introduce a framework to capture a class of assumptions related to subgroup decision. From this class one can use their notation to capture a particular non-interactive assumption. We will be using the syntax and notations from [BWY11].

Let $\mathsf{Bilin\text{-}Gen}_k$ be a PPT algorithm that is parameterized by k, takes as input a security parameter λ and returns $((p_1, p_2, \ldots, p_k), N, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot))$, where p_i's are primes such that $p_i \in \{2^{\lambda-1}, \ldots, 2^{\lambda} - 1\}$ for all $i \in \{1, 2, \ldots, k\}$, $N = \prod_i p_i$, $\mathbb{G}$ and $\mathbb{G}_T$ are groups of order N and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a non-degenerate bilinear map. For any set $S \subseteq \{1, 2, \ldots, k\}$, let $\mathbb{G}_S \subseteq \mathbb{G}$ denote the (unique) subgroup of order $\prod_{i \in S} p_i$. For any $S \subseteq \{1, 2, \ldots, k\}$, using $\{p_i\}_{i \in S}$, one can sample a uniformly random element from $\mathbb{G}_S$.

Assumption 3 (General Subgroup Decision Assumption)**.** Consider the experiment $\mathsf{Expt\text{-}GSD}$, parameterized by $\mathsf{Bilin\text{-}Gen}_k$ and PPT algorithm $\mathcal{A}$, defined in Figure 15.

Experiment $\mathsf{Expt\text{-}GSD}_{\mathsf{Bilin\text{-}Gen}_k, \mathcal{A}}(\lambda)$

1. Challenger chooses $((p_1, \ldots, p_k), N, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot)) \leftarrow \mathsf{Bilin\text{-}Gen}_k(1^{\lambda})$. It sends $(N, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot))$ to the adversary.
2. The adversary sends two subsets $S_0, S_1 \subseteq \{1, 2, \ldots, k\}$. The challenger chooses $b \leftarrow \{0, 1\}$, $g \leftarrow \mathbb{G}_{S_b}$ and sends g to $\mathcal{A}$.
3. The adversary then queries for polynomially many random elements from subgroups of its choice. For each queried set $S \subseteq \{1, 2, \ldots, k\}$ such that either $(S \cap S_0 = S \cap S_1 = \emptyset)$ or $(S \cap S_0 \neq \emptyset$ **and** $S \cap S_1 \neq \emptyset)$, the challenger sends $h \leftarrow \mathbb{G}_S$ to $\mathcal{A}$.
4. Finally, after polynomially many queries, the adversary sends its guess b'. The experiment outputs 1 if $b = b'$, else it outputs 0.

Figure 15: Experiment $\mathsf{Expt\text{-}GSD}$

We say that the General Subgroup Decision holds with respect to $\mathsf{Bilin\text{-}Gen}_k$ if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\text{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\mathsf{Adv}_{\mathcal{A}}(\lambda) = \left|\Pr\left[1 \leftarrow \mathsf{Expt\text{-}GSD}_{\mathsf{Bilin\text{-}Gen}_k, \mathcal{A}}(\lambda)\right] - \frac{1}{2}\right| \leq \text{negl}(\lambda)$.

We will also need an additional assumption on composite order bilinear groups. This assumption is similar to "Assumption 3" of [LW10], but captured in a general framework like [BWY11].

Assumption 4 (Subgroup Decision Assumption in Target Group)**.** Consider the experiment $\mathsf{Expt\text{-}SDT}$, parameterized by $\mathsf{Bilin\text{-}Gen}_k$ and PPT algorithm $\mathcal{A}$, defined in Figure 16.

Experiment $\mathsf{Expt\text{-}SDT}_{\mathsf{Bilin\text{-}Gen}_k,\mathcal{A}}(\lambda)$

1. Challenger chooses $((p_1,\dots,p_k),N,\mathbb{G},\mathbb{G}_T,e(\cdot,\cdot)) \leftarrow \mathsf{Bilin\text{-}Gen}_k(1^\lambda)$. The challenger also chooses $\alpha, s \leftarrow \mathbb{Z}_N$ and $g_1 \leftarrow \mathbb{G}_{\{1\}}$. It sends $(N,\mathbb{G},\mathbb{G}_T,e(\cdot,\cdot))$ to the adversary.
2. The adversary then sends a set $S_1, S_2 \subseteq \{2,\dots,k\}$ such that $S_1 \cap S_2 \neq \emptyset$. The challenger chooses $u, v \leftarrow \mathbb{G}_{S_1}, w \leftarrow \mathbb{G}_{S_2}$. It also chooses $b \leftarrow \{0,1\}$. If $b = 0$, it sets $T = e(g_1,g_1)^{\alpha\cdot s}$, else it sets $T \leftarrow \mathbb{G}_T$. The challenger sends $(g_1^\alpha \cdot u, v, g_1^s \cdot w, T)$. The adversary sends its guess b'. The experiment outputs 1 if $b = b'$, else it outputs 0.

Figure 16: Experiment $\mathsf{Expt\text{-}SDT}$

We say that the Subgroup Decision Assumption in Target Group holds with respect to $\mathsf{Bilin\text{-}Gen}_k$ if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\mathsf{Adv}_{\mathcal{A}}(\lambda) = \left|\Pr\left[1 \leftarrow \mathsf{Expt\text{-}SDT}_{\mathsf{Bilin\text{-}Gen}_k,\mathcal{A}}(\lambda)\right] - \frac{1}{2}\right| \leq \mathrm{negl}(\lambda)$.

Finally, we state the Bilinear Diffie Hellman assumption for composite order groups. This is similar to the BDDH assumption for prime order groups, except that the challenge elements are chosen from a subgroup.

Assumption 5. We say that the Bilinear Decision Diffie Hellman assumption holds with respect to $\mathsf{Bilin\text{-}Gen}_k$ if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\left|\Pr\left[b \leftarrow \mathcal{A}_2(\sigma, N, \mathbb{G}, \mathbb{G}_T, g, h_1, h_2, h_3, U_b) \;:\; \begin{array}{l} ((p_1,\dots,p_k), N, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot)) \leftarrow \mathsf{Bilin\text{-}Gen}_k(1^\lambda) \\ (S,\sigma) \leftarrow \mathcal{A}_1(1^\lambda); g \leftarrow \mathbb{G}_S; a,b,c \leftarrow \mathbb{Z}_N \\ h_1 = g^a, h_2 = g^b, h_3 = g^c \\ U_0 = e(g,g)^{abc}, U_1 \leftarrow \mathbb{G}_T, b \leftarrow \{0,1\} \end{array}\right] - \frac{1}{2}\right|$$

is at most $\mathrm{negl}(\lambda)$.

C Public Key Traitor Tracing Scheme

We will now present our public key traitor tracing scheme over composite order bilinear groups.

C.1 Construction

Let $\mathsf{Bilin\text{-}Gen}_4$ be a group generator that takes as input security parameter λ, parameterized by number of prime-order subgroups $k = 4$, and outputs $((p_1,p_2,p_3,p_4),N,\mathbb{G},\mathbb{G}_T,e(\cdot,\cdot))$, where p_1,p_2,p_3,p_4 are λ-bit primes, $\mathbb{G}$ is a group of order $N = p_1p_2p_3p_4$ with subgroups $\mathbb{G}_{\{1\}},\mathbb{G}_{\{2\}},\mathbb{G}_{\{3\}},\mathbb{G}_{\{4\}}$ of order p_1,p_2,p_3,p_4 respectively. The group $\mathbb{G}_T$ is a target group of order N and e is an efficient (non-degenerate) bilinear function.

$\mathsf{Setup}(1^\lambda, 1^n)$: The setup algorithm first chooses $((p_1,p_2,p_3,p_4),\mathbb{G},\mathbb{G}_T,e(\cdot,\cdot)) \leftarrow \mathsf{Bilin\text{-}Gen}_4(1^\lambda)$. Next, it chooses $g_j \leftarrow \mathbb{G}_{\{j\}}$ for $1 \leq j \leq 4$, $\alpha \leftarrow \mathbb{Z}_N$ and sets $\mathsf{mpk} = (e(g_1,g_1)^\alpha, g_1)$.

The key generation algorithm then chooses an index $i^* \leftarrow \{1,2,\dots,n\}$ and sets the master secret key to be $\mathsf{msk} = (i^*, \alpha, g_1, g_2, g_3, g_4)$.

To choose the secret keys, it uses the $\mathsf{KeyGen}^{\mathsf{less}}$, $\mathsf{KeyGen}^{\mathsf{eq}}$ and $\mathsf{KeyGen}^{\mathsf{gr}}$ algorithms defined below.

$\mathsf{KeyGen}^{\mathsf{less}}(\mathsf{msk})$: It chooses $t, u \leftarrow \mathbb{Z}_N$ and sets $\mathsf{sk} = g_1^\alpha \cdot g_3^t \cdot g_4^u$.

$\mathsf{KeyGen}^{\mathsf{eq}}(\mathsf{msk})$: It chooses $t, u \leftarrow \mathbb{Z}_N$ and sets $\mathsf{sk} = g_1^\alpha \cdot g_2^t \cdot g_4^u$.

$\mathsf{KeyGen}^{\mathsf{gr}}(\mathsf{msk})$: It chooses $u \leftarrow \mathbb{Z}_N$ and sets $\mathsf{sk} = g_1^{\alpha} \cdot g_4^{u}$.

For each $j \in \{1, 2, \ldots, i^*-1\}$, it sets the secret key $\mathsf{sk}_j \leftarrow \mathsf{KeyGen}^{\mathsf{less}}(\mathsf{msk})$. It sets $\mathsf{sk}_{i^*} \leftarrow \mathsf{KeyGen}^{\mathsf{eq}}(\mathsf{msk})$. Finally, for all $j \in \{i^* + 1, \ldots, n\}$, it sets $\mathsf{sk}_j \leftarrow \mathsf{KeyGen}^{\mathsf{gr}}(\mathsf{msk})$.

$\mathsf{Enc}(\mathsf{mpk}, m)$: Let $\mathsf{mpk} = (A, g_1)$. The encryption algorithm first chooses $s \leftarrow \mathbb{Z}_{p_1}$ and sets $\mathsf{ct} = (m \cdot A^s, g_1^s)$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: Let $\mathsf{ct} = (C_0, C_1)$. The decryption algorithm outputs $C_0/e(C_1, \mathsf{sk})$.

$\mathsf{Trace}^D(\mathsf{msk}, 1^y, m_0, m_1)$: Let $\mathsf{msk} = (\alpha, g_1, g_2, g_3, g_4)$ and $\epsilon = \lfloor 1/y \rfloor$.

To define the trace algorithm, we need to first define two encryption algorithms $\mathsf{Enc}^{\mathsf{less}}$ and $\mathsf{Enc}^{\mathsf{leq}}$.

$\mathsf{Enc}^{\mathsf{less}}(\mathsf{msk}, m)$: This encryption algorithm outputs ciphertexts which have group elements from $\mathbb{G}_{\{1,3\}}$. It chooses $s, v \leftarrow \mathbb{Z}_N$ and outputs $\mathsf{ct} = (m \cdot e(g_1, g_1)^{\alpha s}, g_1^s \cdot g_3^v)$.

$\mathsf{Enc}^{\mathsf{leq}}(\mathsf{msk}, m)$: This encryption algorithm outputs ciphertexts which have group elements from $\mathbb{G}_{\{1,2,3\}}$. It chooses $s, u, v \leftarrow \mathbb{Z}_N$ and outputs $\mathsf{ct} = (m \cdot e(g_1, g_1)^{\alpha s}, g_1^s \cdot g_2^u \cdot g_3^v)$.

The trace algorithm first sets $T = \lambda \cdot n/\epsilon$. Let $\mathsf{count}^{\mathsf{leq}} = \mathsf{count}^{\mathsf{less}} = 0$. For $j = 1$ to T, the trace algorithm computes the following:

1. It chooses $b_j \leftarrow \{0, 1\}$ and computes $\mathsf{ct}_j^{\mathsf{less}} \leftarrow \mathsf{Enc}^{\mathsf{less}}(\mathsf{msk}, m_{b_j})$ and sends $\mathsf{ct}_j^{\mathsf{less}}$ to D. If D outputs b_j, set $\mathsf{count}^{\mathsf{less}} = \mathsf{count}^{\mathsf{less}} + 1$, else set $\mathsf{count}^{\mathsf{less}} = \mathsf{count}^{\mathsf{less}} - 1$.
2. It chooses $c_j \leftarrow \{0, 1\}$ and computes $\mathsf{ct}_j^{\mathsf{leq}} \leftarrow \mathsf{Enc}^{\mathsf{less}}(\mathsf{msk}, m_{c_j})$ and sends $\mathsf{ct}_j^{\mathsf{leq}}$ to D. If D outputs c_j, set $\mathsf{count}^{\mathsf{leq}} = \mathsf{count}^{\mathsf{leq}} + 1$, else set $\mathsf{count}^{\mathsf{leq}} = \mathsf{count}^{\mathsf{leq}} - 1$.

If $\mathsf{count}^{\mathsf{less}} - \mathsf{count}^{\mathsf{leq}} > T \cdot (\epsilon/4n)$, output i^*, else output $\perp$.

Correctness Let $\mathsf{msk} = (e(g_1, g_!)^{\alpha}, g_1)$, $\mathsf{msk} = (\alpha, g_1, g_2, g_3, g_4, i)$, $\mathsf{sk}_j = g_1^{\alpha} \cdot g_3^{t_j} \cdot g_4^{u_j}$ for $j < i$, $\mathsf{sk}_i = g_1^{\alpha} \cdot g_2^{t_i} \cdot g_4^{u_i}$ and $\mathsf{sk}_j = g_1^{\alpha} \cdot g_4^{u_j}$ for $j > i$. Fix any message $m \in \mathcal{M}$. The encryption of m with randomness $s \in \mathbb{Z}_N$ is $\mathsf{ct} = (m \cdot e(g_1, g_1)^{\alpha \cdot s}, g_1^s)$. For index $j < i$, $\mathsf{Dec}(\mathsf{sk}_j, \mathsf{ct}) = m \cdot e(g_1, g_1)^{\alpha \cdot s}/e(g_1^{\alpha} \cdot g_3^{t_j} \cdot g_4^{u_j}, g_1^s) = m$. For index i, $\mathsf{Dec}(\mathsf{sk}_i, \mathsf{ct}) = m \cdot e(g_1, g_1)^{\alpha \cdot s}/e(g_1^{\alpha} \cdot g_2^{t_i} \cdot g_4^{u_i}, g_1^s) = m$. For index $j > i$, $\mathsf{Dec}(\mathsf{sk}_j, \mathsf{ct}) = m \cdot e(g_1, g_1)^{\alpha \cdot s}/e(g_1^{\alpha} \cdot g_4^{u_j}, g_1^s) = m$.

Singular Trace Property Note that our scheme satisfies the singular trace property defined in Definition 3.5. The trace algorithm either outputs i (which is chosen during setup, and is part of msk), or outputs $\perp$.

C.2 Proof of Security

C.2.1 IND-CPA Security

First, we will show that the traitor tracing scheme is IND-CPA secure. The following proof is similar to the security proof for El-Gamal encryption scheme.

Theorem C.1. Assuming the Bilinear Decisional Diffie Hellman assumption (Assumption 5), the traitor tracing scheme described above is IND-CPA secure (Definition 3.1).

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ that has advantage ϵ in the IND-CPA security game. Then we can use $\mathcal{A}$ to build a PPT algorithm $\mathcal{B}$ that breaks the BDDH assumption with advantage ϵ.

The reduction algorithm first sends set $S = \{1\}$, and receives $(N, \mathbb{G}, \mathbb{G}_T, e(\cdot, \cdot), g, A, B, C, R)$ from the BDDH challenger, where g is a uniformly random element of $\mathbb{G}_{\{1\}}$, $A = g^a$, $B = g^b$, $C = g^c$ and $R = e(g, g)^{abc}$ or a uniformly random element in $\mathbb{G}_T$. The reduction algorithm implicitly sets $\alpha = ab$ by setting $\mathsf{mpk} = (e(A, B), g)$. Next, it receives messages m_0, m_1 from the adversary $\mathcal{A}$. The reduction algorithm chooses $b \leftarrow \{0,1\}$ and sends $\mathsf{ct} = (m_b \cdot R, C)$ to the adversary. The adversary sends a bit b'. If $b = b'$, the reduction algorithm guesses that $R = e(g,g)^{abc}$, else it guesses that R is uniformly random.

Note that if $R = e(g,g)^{abc}$, then the reduction algorithm perfectly simulates the IND-CPA game. If R is uniformly random, then $\mathcal{A}$ has zero advantage in this game. As a result, if $\mathcal{A}$ can win the IND-CPA game with advantage ϵ, then $\mathcal{B}$ breaks the BDDH assumption with advantage ϵ.

■

C.2.2 False-Trace Probability

Next, we will show that an honest party will not be implicated by our trace algorithm with non-negligible probability.

Theorem C.2. For every PPT adversary $\mathcal{A}$, polynomials $n(\cdot)$, $p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$,

$$\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \leq \mathrm{negl}(\lambda),$$

where $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\cdot)$ is as defined in Definition 3.4.

Proof. Given any pirate decoder box D, let $p_D^{\mathsf{type}} = \Pr[D(\mathsf{ct}) = b : \mathsf{ct} \leftarrow \mathsf{Enc}^{\mathsf{type}}(\mathsf{msk}, m_b)]$ for $\mathsf{type} \in \{\mathsf{less}, \mathsf{leq}\}$, where the probability is taken over bit b, random coins of decoder D and randomness used during encryption. Let $\mathsf{Diff\text{-}Adv}$ denote the event when the advantage of decoder D in distinguishing "less" encryptions of m_0 and m_1 is $\epsilon/8n$ more than its advantage in distinguishing "leq" cipheretexts. Formally, let

$$\mathsf{Diff\text{-}Adv} \ : \ p_D^{\mathsf{less}} - p_D^{\mathsf{leq}} > \epsilon/8n.$$

First, note that we could rewrite the probability of a *false trace* as

$$\Pr[\mathsf{Fal\text{-}Tr}] \leq \Pr[\mathsf{Fal\text{-}Tr} \mid \overline{\mathsf{Diff\text{-}Adv}}] + \Pr[\mathsf{Fal\text{-}Tr} \wedge \mathsf{Diff\text{-}Adv}].$$

To bound the overall probability of a false trace, we start by showing that $\Pr[\mathsf{Fal\text{-}Tr} \mid \overline{\mathsf{Diff\text{-}Adv}}] \leq \mathrm{negl}_1(\lambda)$ by using a simple Chernoff bound. Next, we show that $\Pr[\mathsf{Fal\text{-}Tr} \mid \mathsf{Diff\text{-}Adv}] \leq \mathrm{negl}_2(\lambda)$ by relying on subgroup hiding assumption, i.e. a computational argument. These two lemmas together imply that $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda)$ is also bounded by a negligible function.

Lemma C.1. There exists a negligible function $\mathrm{negl}_1(\cdot)$ such that for every adversary $\mathcal{A}$, all $\lambda \in \mathbb{N}$,

$$\Pr[\mathsf{Fal\text{-}Tr} \mid \overline{\mathsf{Diff\text{-}Adv}}] \leq \mathrm{negl}_1(\lambda).$$

Proof. Consider the binary random variables X^{type} for $\mathsf{type} \in \{\mathsf{less}, \mathsf{leq}\}$ defined as

$$X^{\mathsf{type}} = \begin{cases} 1 & \text{with probability } p_D^{\mathsf{type}}, \\ 0 & \text{otherwise.} \end{cases}$$

Let Z be another random variable defined as $Z = X^{\mathsf{less}} - X^{\mathsf{leq}}$. Now using linearity of expectation, we can write that

$$\mathbb{E}[Z \mid \overline{\mathsf{Diff\text{-}Adv}}] \leq \epsilon/8n.$$

Also, we know that the tracing algorithm estimates $\mathbb{E}[Z]$ by independently sampling $T = \lambda \cdot n/\epsilon$ elements from the distribution induced by Z. In other words, in each trial it first computes a single *less* and *leq* encryptions of messages m_0 and m_1 using uniform randomness, then it uses the pirate box D to decrypt each cipheretext and sets the value of sampled variable appropriately. Let z_i be the sampled value in i^{th} trial. Now we know that $\mathsf{count}^{\mathsf{less}} - \mathsf{count}^{\mathsf{leq}} = \sum_{i=1}^{T} z_i$. Thus, we can write that

$$\Pr[\mathsf{Fal\text{-}Tr} \mid \overline{\mathsf{Diff\text{-}Adv}}] = \Pr\left[\sum_{i=1}^{T} z_i > T \cdot \epsilon/4n \mid \overline{\mathsf{Diff\text{-}Adv}}\right].$$

Using a Chernoff bound, we can bound the above probability as

$$\Pr[\mathsf{Fal\text{-}Tr} \mid \overline{\mathsf{Diff\text{-}Adv}}] \leq e^{-T\epsilon/16n}.$$

Substituting $T = \lambda \cdot n/\epsilon$, we get $\Pr[\mathsf{Fal\text{-}Tr} \mid \overline{\mathsf{Diff\text{-}Adv}}] \leq 2^{-O(\lambda)} = \mathrm{negl}(\lambda)$. This completes the proof. ∎

Lemma C.2. Assuming the subgroup hiding assumption (Assumption 3), for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$,

$$\Pr[\mathsf{Fal\text{-}Tr} \wedge \mathsf{Diff\text{-}Adv}] \leq \mathrm{negl}_2(\lambda).$$

Here, the events $\mathsf{Fal\text{-}Tr}$ and $\mathsf{Diff\text{-}Adv}$ are parameterized by the adversary $\mathcal{A}$.

Proof. Suppose, on the contrary, there exists a PPT adversary $\mathcal{A}$ such that $\Pr[\mathsf{Fal\text{-}Tr} \wedge \mathsf{Diff\text{-}Adv}] = \eta(\lambda)$, where η is non-negligible. Then we can construct a reduction algorithm $\mathcal{B}$ that breaks the subgroup hiding assumption.

The reduction algorithm $\mathcal{B}$ sends challenge sets $S_0 = \{3\}$ and $S_1 = \{2, 3\}$ and gets group element T, together with $N, \mathbb{G}, \mathbb{G}_T, e(\cdot, \cdot)$, where $T \in \mathbb{G}_{S_0}$ or $T \in \mathbb{G}_{S_1}$. Next, it queries for generators of $\mathbb{G}_{\{1\}}, \mathbb{G}_{\{3\}}$ and $G_{\{4\}}$, and receives g_1, g_3, g_4 respectively.

It first chooses $\alpha \leftarrow \mathbb{Z}_N$, chooses $i \leftarrow \{1, 2, \ldots, n\}$ and computes mpk using g_1, α. Next, it computes the secret keys using g_1, g_3, g_4. Concretely, if it is queried for secret key corresponding to i, then the reduction algorithm outputs a uniformly random bit and quits. (hence it does not require generator for $\mathbb{G}_{\{2\}}$). For $j < i$, it chooses $t_j, u_j \leftarrow \mathbb{Z}_N$ and sets $\mathsf{sk}_j = g_1^{\alpha} \cdot g_3^{t_j} \cdot g_4^{u_j}$. For $j > i$, it chooses $t_j \leftarrow \mathbb{Z}_N$ and sets $\mathsf{sk}_j = g_1^{\alpha} \cdot g_4^{t_j}$. Finally, after all queries, the adversary sends a pirate box D and messages m_0, m_1.

The reduction algorithm first computes an estimate of $\mathsf{Diff\text{-}Adv}$. It sets $\mathsf{count}^{\mathsf{less}} = \mathsf{count} = 0$ and $z = \lambda \cdot n/\epsilon$. For $k = 1$ to z, it does the following: it chooses $s_k, v_k \leftarrow \mathbb{Z}_N$ and $b_k \leftarrow \{0, 1\}$, sets $\mathsf{ct}_k = (m_{b_k} \cdot e(g_1^{\alpha}, g_1^{s_k}), g_1^{s_k} \cdot T^{v_k})$. If $D(\mathsf{ct}_k) = b_k$, it sets $\mathsf{count} = \mathsf{count} + 1$, else it sets $\mathsf{count} = \mathsf{count} - 1$. Next, it chooses $s'_k, v'_k \leftarrow \mathbb{Z}_N$ and $b'_k \leftarrow \{0, 1\}$, sets $\mathsf{ct}'_k = (m_{b'_k} \cdot e(g_1^{\alpha}, g_1^{s'_k}), g_1^{s'_k} \cdot g_3^{v'_k})$. If $D(\mathsf{ct}'_k) = b'_k$, it sets $\mathsf{count}^{\mathsf{less}} = \mathsf{count}^{\mathsf{less}} + 1$, else it sets $\mathsf{count}^{\mathsf{less}} = \mathsf{count}^{\mathsf{less}} - 1$. Finally, if $\mathsf{count}^{\mathsf{less}} - \mathsf{count} > \epsilon/16n$, $\mathcal{B}$ guesses that $T \in \mathbb{G}_{S_1}$, else it guesses that $T \in \mathbb{G}_{S_0}$.

First, note that $\Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_1} \mid T \in \mathbb{G}_{S_0}] \leq (1/2) \Pr[\mathcal{A} \text{ sends } i \text{ as query } \mid T \in \mathbb{G}_{S_0}] + \mathrm{negl}(\lambda)$. This is because if $\mathcal{A}$ does not query for index i and $T \in \mathbb{G}_{S_0}$, then the expected value of $\mathsf{count}^{\mathsf{less}} - \mathsf{count} = 0$. As a result, using Chernoff bounds, we can argue that the probability that $\Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_1} \mid T \in \mathbb{G}_{S_0} \wedge i \text{ not queried}] \leq 2^{-O(\lambda)}$. Also, $\Pr[\mathcal{A} \text{ sends } i \text{ as query } \mid T \in \mathbb{G}_{S_b}] = \Pr[\mathcal{A} \text{ sends } i \text{ as query }]$, i.e. it is independent of bit b, becasue the adversary $\mathcal{A}$ does not receive T. Thus, $\Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_1} \mid T \in \mathbb{G}_{S_0}] \leq (1/2) \Pr[\mathcal{A} \text{ sends } i \text{ as query}] + \mathrm{negl}(\lambda)$. Below we compute $\Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_1} \mid T \in \mathbb{G}_{S_1}]$.

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_1} \mid T \in \mathbb{G}_{S_1}] \\
&\quad = \frac{1}{2} \Pr[\mathcal{A} \text{ sends } i \text{ as query } \mid T \in \mathbb{G}_{S_1}] \\
&\qquad + \Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_1} \wedge \mathcal{A} \text{ does not send } i \text{ as query } \mid T \in \mathbb{G}_{S_1}]
\end{aligned}$$

Now from construction of our reduction algorithm, we know that

$$\Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_1} \wedge \mathcal{A} \text{ does not send } i \text{ as query } \mid T \in \mathbb{G}_{S_1}] \geq \Pr[\mathsf{Fal\text{-}Tr} \wedge \mathsf{Diff\text{-}Adv}].$$

Therefore, the reduction algorithm $\mathcal{B}$'s advantage could be written as

$$\begin{aligned}
\Pr[\mathcal{B} \text{ wins}] &= \Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_1} \mid T \in \mathbb{G}_{S_1}] - \Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_1} \mid T \in \mathbb{G}_{S_0}] \\
&\geq \Pr[\mathsf{Fal\text{-}Tr} \wedge \mathsf{Diff\text{-}Adv}] - \mathrm{negl}(\lambda) \geq \eta(\lambda) - \mathrm{negl}(\lambda).
\end{aligned}$$

This concludes our proof. ∎

∎

C.2.3 Correct-Trace Probability

We will first introduce some notations for our security proof. For any $\gamma \in [0, 1/2]$, a decoding box D is said to be γ-Dist for messages m_0, m_1 if

$$\Pr[D(\mathsf{ct}) = b \ : \ b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, m_b)] \geq 1/2 + \gamma.$$

Similarly, a decoding box D is said to be γ-$\mathsf{Dist}^{\mathsf{less}}$ for messages m_0, m_1 if

$$\Pr[D(\mathsf{ct}) = b \ : \ b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}^{\mathsf{less}}(\mathsf{msk}, m_b)] \geq 1/2 + \gamma.$$

Finally, we say that D is γ-$\mathsf{Dist}^{\mathsf{leq}}$ for messages m_0, m_1 if

$$\Pr[D(\mathsf{ct}) = b \ : \ b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}^{\mathsf{leq}}(\mathsf{msk}, m_b)] \geq 1/2 + \gamma.$$

For any adversary $\mathcal{A}$ and polynomial $n(\cdot)$, we define experiment $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)$ (see Figure 17). The experiment takes as input a security parameter λ, index $i \in \{1, 2, \ldots, n(\lambda)\}$ and outputs a decoding box D and two messages m_0, m_1.

Experiment $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)$

1. Challenger chooses $((p_1, p_2, p_3, p_4), N, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot)) \leftarrow \mathsf{Bilin\text{-}Gen}_4(1^\lambda)$, $g_j \leftarrow \mathbb{G}_{\{j\}}$ for $j \in \{1,2,3,4\}$ and $\alpha \leftarrow \mathbb{Z}_N$. It sets $\mathsf{mpk} = (e(g_1, g_1)^\alpha, g_1)$ and sends mpk to $\mathcal{A}$.
2. The set S represents the keys queried by $\mathcal{A}$. For each queried index $j \in S$, if $j < i$, the challenger computes $\mathsf{sk}_j \leftarrow \mathsf{KeyGen}^{\mathsf{less}}(\mathsf{msk})$; if $j = i$, the challenger computes $\mathsf{sk}_i \leftarrow \mathsf{KeyGen}^{\mathsf{eq}}(\mathsf{msk})$; else, for $j > i$, it computes $\mathsf{sk}_j \leftarrow \mathsf{KeyGen}^{\mathsf{gr}}(\mathsf{msk})$ and sends sk_j.
3. The adversary finally outputs a pirate decoding box D and messages m_0, m_1. The output of the experiment is (D, m_0, m_1).

Figure 17: Experiment $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)$

Using the $\mathsf{MakeBox}$ experiment, we can define the following probabilities, parameterized by $\gamma \in [0, 1/2]$, and a function of λ, i:

$$\Pr\text{-}\mathsf{Good\text{-}Dec}_{\mathcal{A},n,\gamma}(\lambda, i) = \Pr\left[D \text{ is } \gamma\text{-}\mathsf{Dist} \text{ for } m_0, m_1 \ : \ (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)\right]$$

$$\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{leq}}_{\mathcal{A},n,\gamma}(\lambda, i) = \Pr\left[D \text{ is } \gamma\text{-}\mathsf{Dist}^{\mathsf{leq}} \text{ for } m_0, m_1 \ : \ (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)\right]$$

$$\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\mathcal{A},n,\gamma}(\lambda, i) = \Pr\left[D \text{ is } \gamma\text{-}\mathsf{Dist}^{\mathsf{less}} \text{ for } m_0, m_1 \ : \ (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)\right]$$

$$\Pr\text{-}\mathsf{Gap}_{\mathcal{A},n,\gamma}(\lambda, i) = \Pr\left[\exists\ \delta \in [0, 1/2] \text{ s.t. } \begin{array}{l} D \text{ is } \delta\text{-}\mathsf{Dist}^{\mathsf{less}} \wedge \\ D \text{ is not } (\delta - \gamma)\text{-}\mathsf{Dist}^{\mathsf{leq}} \end{array} \ : \ (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)\right]$$

These probabilities are defined over all the random coins chosen by the challenger and adversary $\mathcal{A}$ during $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)$ experiment.

First, we will show that $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}$ is related to $\Pr\text{-}\mathsf{Gap}$ via the following relation.

Theorem C.3. Let $\mathcal{A}$ be a PPT adversary, $n(\cdot), p(\cdot)$ polynomials and $\epsilon(\cdot)$ a non-negligible function. There exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$,

$$\sum_i \Pr\text{-}\mathsf{Gap}_{\mathcal{A},n,\epsilon/4n}(\lambda, i) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) - \mathrm{negl}(\lambda).$$

Next, we will show that $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}$ is related to $\Pr\text{-}\mathsf{Gap}$ via the following relation.

Theorem C.4. Let $\mathcal{A}$ be a PPT adversary, $n(\cdot), p(\cdot)$ polynomials and $\epsilon(\cdot)$ a non-negligible function. There exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$,

$$\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \geq \frac{\left(\sum_i \Pr\text{-}\mathsf{Gap}_{\mathcal{A},n,\epsilon/4n}(\lambda, i)\right)}{n(\lambda)} - \mathrm{negl}(\lambda).$$

Observe that combing above two theorems, we get that our scheme is a $1/n$-risky secure traitor tracing scheme. We will now prove these theorems in the following subsections.

C.2.4 Proof of Theorem C.3

For notational simplicity, we will skip the dependence of n and ϵ on λ. Also, we will skip the subscripts $\mathcal{A}$ and n when they are clear from the context.

Outline of the proof. At a high level, this proof can be divided into the following steps:

- We first show that $\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon/2n}(1) \approx \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}$ and $\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon(n+1)/2n}(n+1) \approx 0$ (see Observation C.1, Lemma C.3 and Lemma C.4).
- From this, it follows that $\exists\ \Gamma \subseteq \{1, 2, \ldots, n\}$ such that for all $i \in \Gamma$, $\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot i/2n}(i) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot(i+1)/2n}(i+1) > 0$ and the sum of these differences is at least $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon} - \mathrm{negl}$ (see Observation C.2).
- Next, we show $\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{leq}}_{\epsilon-\epsilon\cdot(i+1)/2n+\epsilon/4n}(i) \approx \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot(i+1)/2n}(i+1)$ (see Lemma C.5).
- After this, we relate $\Pr\text{-}\mathsf{Gap}(i)$ to $\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}(i)$ and $\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{leq}}(i)$. We show
$$\begin{aligned}\Pr\text{-}\mathsf{Gap}_{\epsilon/4n}(i) &\geq \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot i/2n}(i) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{leq}}_{\epsilon-\epsilon\cdot(i+1)/2n+\epsilon/4n}(i) \text{ (see Lemma C.6)}\\ &\approx \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot i/2n}(i) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot(i+1)/2n}(i+1)\end{aligned}$$
- As a result, we can conclude that
$$\begin{aligned}\sum_i \Pr\text{-}\mathsf{Gap}_{\epsilon/4n}(i) &\geq \sum_{i\in\Gamma} \Pr\text{-}\mathsf{Gap}_{\epsilon/4n}(i)\\ &\geq \sum_{i\in\Gamma}\left(\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot i/2n}(i) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot(i+1)/2n}(i+1)\right)\\ &\geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon} - \mathrm{negl}\end{aligned}$$

First, we have the following observation.

Observation C.1. For every adversary $\mathcal{A}$, polynomial $n(\cdot)$ and $\lambda \in \mathbb{N}$, there exists an $i^* \in \{1, 2, \ldots, n(\lambda)\}$ such that $\Pr\text{-}\mathsf{Good\text{-}Dec}_{\mathcal{A},n,\epsilon}(\lambda, i^*) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda)$.

This observation simply follows from the fact that $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) = (1/n)\sum_i \Pr\text{-}\mathsf{Good\text{-}Dec}_{\mathcal{A},n,\epsilon}(\lambda, i)$, and therefore, there exists some index i^* such that $\Pr\text{-}\mathsf{Good\text{-}Dec}_{\mathcal{A},n,\epsilon}(\lambda, i^*) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda)$.

Lemma C.3. Assuming the subgroup decision assumption (Assumption 3), for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$,

$$\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon/2n}(\lambda, 1) \geq \Pr\text{-}\mathsf{Good\text{-}Dec}_{\epsilon}(\lambda, i^*) - \mathrm{negl}(\lambda) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) - \mathrm{negl}(\lambda).$$

Proof. Let $\rho_1(\lambda) = \Pr\text{-}\mathsf{Good\text{-}Dec}_{\epsilon}(\lambda, i^*)$ and $\rho_2(\lambda) = \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon/2n}(\lambda, 1)$. In order to show that $\rho_1 - \rho_2$ is negligible in λ, we will define a sequence of hybrid events, and show that the difference in their probabilities is at most negligible in λ.

Hybrid Hyb_1 This experiment is similar to that associated with $\mathsf{Good\text{-}Decoder}_\epsilon$, except that all keys for indices $j < i^*$ are generated using $\mathsf{KeyGen}^{\mathsf{gr}}$. The adversary must finally output a pirate box D and two messages m_0, m_1 such that D can distinguish between (normal) encryptions of m_0 and m_1 with advantage at least $\gamma_1 = \epsilon - \epsilon/8n$. Let $\rho_{\mathsf{Hyb}_1}(\lambda)$ denote the probability that $\mathcal{A}$ outputs a pirate box D that is γ_1-Dist for m_0, m_1.

Hybrid Hyb_2 This experiment is similar to that associated witho $\mathsf{Good\text{-}Decoder}_\epsilon$, except that the keys corresponding to index i^* is also generated using $\mathsf{KeyGen}^{\mathsf{gr}}$. The adversary must finally output a pirate box D and two messages m_0, m_1 such that D can distinguish between (normal) encryptions of m_0 and m_1 with advantage at least $\gamma_2 = \epsilon - \epsilon/6n$. Let $\rho_{\mathsf{Hyb}_2}(\lambda)$ denote the probability that $\mathcal{A}$ outputs a pirate box D that is γ_2-Dist for m_0, m_1.

Hybrid Hyb_3 This experiment is similar to that associated with Hyb_2, except that the key for index $i = 1$ is generated using $\mathsf{KeyGen}^{\mathsf{eq}}$; all other keys are generated using $\mathsf{KeyGen}^{\mathsf{gr}}$. The adversary must finally output a pirate box D and two messages m_0, m_1 such that D can distinguish between (normal) encryptions of m_0 and m_1 with advantage at least $\gamma_3 = \epsilon - \epsilon/4n$. Let $\rho_{\mathsf{Hyb}_3}(\lambda)$ denote the probability that $\mathcal{A}$ outputs a pirate box D that is γ_3-Dist for m_0, m_1.

Claim C.1. Assuming the subgroup decision assumption (Assumption 3), for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}_1(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_1(\lambda) - \rho_{\mathsf{Hyb}_1}(\lambda) \leq \text{negl}_1(\lambda)$.

Proof. Suppose $\rho_1 - \rho_{\mathsf{Hyb}_1} = \eta$ for some non-negligible function η. We will construct a reduction algorithm $\mathcal{B}$ that breaks the subgroup decision assumption with non-negligible advantage.

First, $\mathcal{B}$ sends its challenge sets $S_0 = \{3, 4\}$, $S_1 = \{4\}$ and it receives T, where $T \in \mathbb{G}_{S_0}$ or $T \in \mathbb{G}_{S_1}$. Next, it queries for the generators for $\mathbb{G}_{\{1\}}, \mathbb{G}_{\{2\}}$ and $\mathbb{G}_{\{4\}}$, and receives g_1, g_2, g_4 respectively. The reduction algorithm first chooses $\alpha \leftarrow \mathbb{Z}_N$ and sets $\mathsf{mpk} = (e(g_1, g_1)^\alpha, g_1)$. Next, it uses g_1 and T to construct keys for indices less than i^*; that is, it chooses $u_j \leftarrow \mathbb{Z}_N$ and sets $\mathsf{sk}_j = g_1^\alpha \cdot T^{u_j}$ as the secret key for index $j < i^*$. For the i^* index, it uses g_1, g_2 and g_4; that is, it chooses u_{i^*}, t_{i^*} and sets $\mathsf{sk}_{i^*} = g_1^\alpha \cdot g_2^{u_{i^*}} \cdot g_4^{t_{i^*}}$. Finally, for indices $j > i^*$, the reduction algorithm uses g_1 and g_4 and sets $\mathsf{sk}_j = g_1^\alpha \cdot g_4^{u_j}$ for randomly chosen $u_j \leftarrow \mathbb{Z}_N$.

After all secret key queries, the reduction algorithm receives pirate box D and m_0, m_1. The reduction algorithm sets $\gamma = \epsilon - \epsilon/16n$, $z = \lambda \cdot n/\epsilon$ and tests whether D is a γ-Dist box for m_0, m_1 using simple counting based estimation. Concretely, it first sets $\mathsf{count} = 0$. For $k = 1$ to z, it chooses $b_k \leftarrow \{0, 1\}$, $s_k \leftarrow \mathbb{Z}_N$ and sets $\mathsf{ct}_k = (m_{b_k} \cdot e(g_1, g_1)^{\alpha \cdot s_k}, g_1^{s_k})$. Next, if $D(\mathsf{ct}_k) = b_k$, it sets $\mathsf{count} = \mathsf{count} + 1$, else it sets $\mathsf{count} = \mathsf{count} - 1$. Finally, after the z iterations, if $\mathsf{count} > \gamma \cdot z$, then $\mathcal{B}$ guesses that $T \in \mathbb{G}_{S_0}$, else it guesses that $T \in \mathbb{G}_{S_1}$.

Let us now compute the reduction algorithm's advantage. First, note that if $T \leftarrow \mathbb{G}_{S_0}$ then the key for indices $j < i^*$ corresponds to a $\mathsf{KeyGen}^{\mathsf{less}}$ key, and if $T \in \mathbb{G}_{S_1}$, then it corresponds to a $\mathsf{KeyGen}^{\mathsf{gr}}$ key (we use the Chinese Remainder Theorem to argue that $\{(g_1^\alpha \cdot g_{3,4}^{u_j})_j \ : \ g_1 \leftarrow \mathbb{G}_1, g_{3,4} \leftarrow \mathbb{G}_{\{3,4\}}, u_j \leftarrow \mathbb{Z}_N\}$ is statistically indistinguishable from $\{(g_1^\alpha \cdot g_3^{u_j} \cdot g_4^{v_j}) \ : \ g_1 \leftarrow \mathbb{G}_{\{1\}}, g_3 \leftarrow \mathbb{G}_{\{3\}}, g_4 \leftarrow \mathbb{G}_{\{4\}}, u_j \leftarrow \mathbb{Z}_N, v_j \leftarrow \mathbb{Z}_N\}$).

Similarly, using Chinese Remainder Theorem, we can argue that the ciphertexts computed by the reduction algorithm are indistinguishable from (normal) ciphertexts. We will now analyse $\mathcal{B}'s$ advantage in breaking the subgroup decision assumption.

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_0} \mid T \in \mathbb{G}_{S_0}] \\
&\quad = \Pr[\mathsf{count} > \gamma \cdot z \mid T \in \mathbb{G}_{S_0}] \\
&\quad \geq \Pr[\mathcal{A} \text{ outputs } \epsilon\text{-}\mathsf{Dist} \text{ box } D \wedge \mathsf{count} > \gamma \cdot z \mid T \in \mathbb{G}_{S_0}] \\
&\quad \geq \Pr[\mathcal{A} \text{ outputs } \epsilon\text{-}\mathsf{Dist} \text{ box } D \mid T \in \mathbb{G}_{S_0}] \\
&\quad\ - \Pr[\mathcal{A} \text{ outputs } \epsilon\text{-}\mathsf{Dist} \text{ box } D \wedge \mathsf{count} \leq \gamma \cdot z \mid T \in \mathbb{G}_{S_0}] \\
&\quad \geq \Pr\text{-}\mathsf{Good\text{-}Dec}_{\epsilon}(\lambda, i^*) \\
&\quad\ - \Pr[\mathsf{count} \leq \gamma \cdot z \mid T \in \mathbb{G}_{S_0} \wedge \mathcal{A} \text{ outputs } \epsilon\text{-}\mathsf{Dist} \text{ box } D] \\
&\quad \geq \rho_1 - 2^{-O(\lambda)}.
\end{aligned}$$

The last inequality follows by applying a Chernoff bound similar to that in Lemma C.1. Next, let us analyse the probability that $\mathcal{B}$ guesses $T \in \mathbb{G}_{S_0}$ when $T \in \mathbb{G}_{S_1}$.

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_0} \mid T \in \mathbb{G}_{S_1}] \\
&\quad = \Pr[\mathsf{count} > \gamma \cdot z \mid T \in \mathbb{G}_{S_1}] \\
&\quad \leq \Pr[\mathcal{A} \text{ does not output } (\epsilon - \epsilon/8n)\text{-}\mathsf{Dist} \text{ box } D \wedge \mathsf{count} > \gamma \cdot z \mid T \in \mathbb{G}_{S_1}] \\
&\quad\ + \Pr[\mathcal{A} \text{ outputs } (\epsilon - \epsilon/8n)\text{-}\mathsf{Dist} \text{ box } D \wedge \mathsf{count} > \gamma \cdot z \mid T \in \mathbb{G}_{S_1}] \\
&\quad \leq \Pr[\mathsf{count} > \gamma \cdot z \mid T \in \mathbb{G}_{S_1} \wedge \mathcal{A} \text{ does not output } (\epsilon - \epsilon/8n)\text{-}\mathsf{Dist} \text{ box } D] + \rho_{\mathsf{Hyb}_1} \\
&\quad \leq\ 2^{-O(\lambda)} + \rho_{\mathsf{Hyb}_1}.
\end{aligned}$$

As before, the last inequality follows by applying a Chernoff bound. Thus, combining above bounds, we get that

$$\begin{aligned}
\Pr[\mathcal{B} \text{ wins}] &= \Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_0} \mid T \in \mathbb{G}_{S_0}] - \Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_0} \mid T \in \mathbb{G}_{S_1}] \\
&\geq \rho_1 - \rho_{\mathsf{Hyb}_1} - \mathrm{negl}(\lambda).
\end{aligned}$$

Therefore, since the subgroup decision assumption holds over $\mathbb{G}$, thus we can conclude that $\rho_1 - \rho_{\mathsf{Hyb}_1} \leq \mathrm{negl}_1(\lambda)$ for some negligible function $\mathrm{negl}(\cdot)$. ■

Claim C.2. Assuming the subgroup decision assumption (Assumption 3), for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_{\mathsf{Hyb}_1}(\lambda) - \rho_{\mathsf{Hyb}_2}(\lambda) \leq \mathrm{negl}_2(\lambda)$.

Proof. Suppose $\rho_{\mathsf{Hyb}_1} - \rho_{\mathsf{Hyb}_2} = \eta$ for some non-negligible function η. We will construct a reduction algorithm $\mathcal{B}$ that breaks the subgroup decision assumption with non-negligible advantage. The proof of this claim is similar to that of Claim C.1.

First, $\mathcal{B}$ sends its challenge sets $S_0 = \{2, 4\}$, $S_1 = \{4\}$ and it receives T, where $T \in \mathbb{G}_{S_0}$ or $T \in \mathbb{G}_{S_1}$. Next, it queries for the generators for $\mathbb{G}_{\{1\}}, \mathbb{G}_{\{3\}}$ and $\mathbb{G}_{\{4\}}$, and receives g_1, g_3, g_4 respectively. The reduction algorithm first chooses $\alpha \leftarrow \mathbb{Z}_N$ and sets $\mathsf{mpk} = (e(g_1, g_1)^{\alpha}, g_1)$. Next, it uses g_1 and g_4 to construct keys for indices less than i^*; that is, it chooses $u_j \leftarrow \mathbb{Z}_N$ and sets $\mathsf{sk}_j = g_1^{\alpha} \cdot g_4^{u_j}$ as the secret key for index $j < i^*$. For the i^* index, it uses g_1 and T; that is, it chooses u_{i^*} and sets $\mathsf{sk}_{i^*} = g_1^{\alpha} \cdot T^{u_{i^*}}$. Finally, for indices $j > i^*$, the reduction algorithm uses g_1 and g_4 and sets $\mathsf{sk}_j = g_1^{\alpha} \cdot g_4^{u_j}$ for randomly chosen $u_j \leftarrow \mathbb{Z}_N$.

After all secret key queries, the reduction algorithm receives pirate box D and m_0, m_1. The reduction algorithm sets $\gamma = \epsilon - \epsilon/7n$, $z = \lambda \cdot n/\epsilon$, $\mathsf{count} = 0$. Next, it tests whether D is a γ-Dist box for m_0, m_1. For $k = 1$ to z, it chooses $b_k \leftarrow \{0, 1\}$, $s_k \leftarrow \mathbb{Z}_N$, sets $\mathsf{ct}_k = (m_{b_k} \cdot e(g_1, g_1)^{\alpha \cdot s_k}, g_1^{s_k})$ and if $D(\mathsf{ct}_k) = b_k$, it sets $\mathsf{count} = \mathsf{count} + 1$, else it sets $\mathsf{count} = \mathsf{count} - 1$. Finally, after the z iterations, if $\mathsf{count} > \gamma \cdot z$, then $\mathcal{B}$ guesses that $T \in \mathbb{G}_{S_0}$, else it guesses that $T \in \mathbb{G}_{S_1}$.

The analysis of $\mathcal{B}$'s advantage is similar to that in proof of Claim C.1. ■

Claim C.3. Assuming the subgroup decision assumption (Assumption 3), for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}_3(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_{\mathsf{Hyb}_2}(\lambda) - \rho_{\mathsf{Hyb}_3}(\lambda) \leq \text{negl}_3(\lambda)$.

Proof. Suppose $\rho_{\mathsf{Hyb}_2} - \rho_{\mathsf{Hyb}_3} = \eta$ for some non-negligible function η. We will construct a reduction algorithm $\mathcal{B}$ that breaks the subgroup decision assumption with non-negligible advantage. The proof of this claim is similar to that of Claim C.1.

First, $\mathcal{B}$ sends its challenge sets $S_0 = \{4\}$, $S_1 = \{2, 4\}$ and it receives T, where $T \in \mathbb{G}_{S_0}$ or $T \in \mathbb{G}_{S_1}$. Next, it queries for the generators for $\mathbb{G}_{\{1\}}, \mathbb{G}_{\{3\}}$ and $\mathbb{G}_{\{4\}}$, and receives g_1, g_3, g_4 respectively. The reduction algorithm first chooses $\alpha \leftarrow \mathbb{Z}_N$ and sets $\mathsf{mpk} = (e(g_1, g_1)^\alpha, g_1)$. Next, it uses g_1 and T to construct the secret key for $i = 1$. It sets $\mathsf{sk}_1 = (g_1^\alpha \cdot T)$. Finally, for indices $j > 1$, the reduction algorithm uses g_1 and g_4 and sets $\mathsf{sk}_j = g_1^\alpha \cdot g_4^{u_j}$ for randomly chosen $u_j \leftarrow \mathbb{Z}_N$.

After all secret key queries, the reduction algorithm receives pirate box D and m_0, m_1. The reduction algorithm sets $\gamma = \epsilon - \epsilon/5n$, $z = \lambda \cdot n/\epsilon$, $\mathsf{count} = 0$. Next, it tests whether D is a γ-Dist box for m_0, m_1. For $k = 1$ to z, it chooses $b_k \leftarrow \{0, 1\}$, $s_k \leftarrow \mathbb{Z}_N$, sets $\mathsf{ct}_k = (m_{b_k} \cdot e(g_1, g_1)^{\alpha \cdot s_k}, g_1^{s_k})$ and if $D(\mathsf{ct}_k) = b_k$, it sets $\mathsf{count} = \mathsf{count} + 1$, else it sets $\mathsf{count} = \mathsf{count} - 1$. Finally, after the z iterations, if $\mathsf{count} > \gamma \cdot z$, then $\mathcal{B}$ guesses that $T \in \mathbb{G}_{S_0}$, else it guesses that $T \in \mathbb{G}_{S_1}$.

The analysis of $\mathcal{B}$'s advantage is similar to that in proof of Claim C.1. ■

Claim C.4. Assuming the subgroup decision assumption (Assumption 3), for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}_3(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_{\mathsf{Hyb}_3}(\lambda) - \rho_2(\lambda) \leq \text{negl}_3(\lambda)$.

Proof. Suppose $\rho_{\mathsf{Hyb}_3} - \rho_2 = \eta$ for some non-negligible function η. We will construct a reduction algorithm $\mathcal{B}$ that breaks the subgroup decision assumption with non-negligible advantage. The proof of this claim is similar to that of Claim C.1.

First, $\mathcal{B}$ sends its challenge sets $S_0 = \{1\}$, $S_1 = \{1, 3\}$ and it receives T, where $T \in \mathbb{G}_{S_0}$ or $T \in \mathbb{G}_{S_1}$. Next, it queries for the generators for $\mathbb{G}_{\{1\}}$, $\mathbb{G}_{\{2\}}$ and $\mathbb{G}_{\{4\}}$, and receives g_1, g_2, g_4 respectively. The reduction algorithm first chooses $\alpha \leftarrow \mathbb{Z}_N$ and sets $\mathsf{mpk} = (e(g_1, g_1)^\alpha, g_1)$. Next, it uses g_1, g_2 and g_4 to construct keys for index 1; that is, it chooses $t_1, u_1 \leftarrow \mathbb{Z}_N$ and sets $\mathsf{sk}_1 = g_1^\alpha \cdot g_2^{t_1} \cdot g_4^{u_1}$. For indices $j > 1$, the reduction algorithm uses g_1 and g_4 and sets $\mathsf{sk}_j = g_1^\alpha \cdot g_4^{u_j}$ for randomly chosen $u_j \leftarrow \mathbb{Z}_N$.

After all secret key queries, the reduction algorithm receives pirate box D and m_0, m_1. The reduction algorithm sets $\gamma = \epsilon - \epsilon/3n$, $z = \lambda \cdot n/\epsilon$, $\mathsf{count} = 0$. Next, it tests whether D is a γ-Dist box for m_0, m_1. For $k = 1$ to z, it chooses $b_k \leftarrow \{0, 1\}$, $s_k \leftarrow \mathbb{Z}_N$, sets $\mathsf{ct}_k = (m_{b_k} \cdot e(g_1, T)^{\alpha \cdot s_k}, T^{s_k})$ and if $D(\mathsf{ct}_k) = b_k$, it sets $\mathsf{count} = \mathsf{count} + 1$, else it sets $\mathsf{count} = \mathsf{count} - 1$. Finally, after the z iterations, if $\mathsf{count} > \gamma \cdot z$, then $\mathcal{B}$ guesses that $T \in \mathbb{G}_{S_0}$, else it guesses that $T \in \mathbb{G}_{S_1}$.

The analysis of $\mathcal{B}$'s advantage is similar to that in proof of Claim C.1. ■

■

Lemma C.4. Assuming the subgroup hiding in target group assumption (Assumption 4), for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$,

$$\Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\mathcal{A}, \epsilon - \frac{\epsilon \cdot (n+1)}{2n}}(\lambda, n+1) \leq \text{negl}(\lambda).$$

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ such that $\rho = \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\mathcal{A}, \epsilon - \frac{\epsilon \cdot (n+1)}{2n}}(\lambda, n+1)$ is non-negligible in λ. The adversary $\mathcal{A}$ receives keys generated by $\mathsf{KeyGen}^{\mathsf{less}}$ (that is, all the secret keys have $\mathbb{G}_{\{1,3,4\}}$ group elements), and must output a pirate box D and messages m_0, m_1 such that D can distinguish between encryptions of m_0 and m_1 generated using $\mathsf{Enc}^{\mathsf{less}}$ with probability at least $\kappa = \epsilon - \frac{\epsilon \cdot (n+1)}{2n}$. We will show that $\mathcal{A}$ can be used to build a PPT algorithm $\mathcal{B}$ that breaks Assumption 4 with non-negligible advantage.

The reduction algorithm first sends $S_1 = \{3,4\}$ and $S_2 = \{3\}$. It receives $(g_1,\ h_1,\ h_2,\ h_3,\ T)$ from the challenger, where $h_1 = g_1^{\alpha} \cdot u$, $h_3 = g_1^{s} \cdot w$, $g_1 \in \mathbb{G}_{\{1\}}$, $u, h_2 \in \mathbb{G}_{S_1}$, $w \in \mathbb{G}_{S_2}$ and T is either $e(g_1, g_1)^{\alpha \cdot s}$ or a uniformly random element in $\mathbb{G}_T$. It sets $\mathsf{mpk} = (e(g_1, h_1), g_1)$. It responds to the secret key queries using h_1, h_2; that is, the secret key for j is $\mathsf{sk}_j = (h_1 \cdot h_2^{u_j})$ for some randomly chosen u_j. Note that elements from $\mathbb{G}_{\{2\}}$ are not required since all keys are created using $\mathsf{KeyGen}^{\mathsf{less}}$.

Finally, it receives a pirate box D and m_0, m_1. The reduction algorithm sets $\gamma = \epsilon/8$ and $\mathsf{count} = 0$. Next, it tests whether D is a γ-$\mathsf{Dist}^{\mathsf{less}}$ box for m_0, m_1. Concretely, for $k = 1$ to λ/ϵ, it chooses $b_k \leftarrow \{0,1\}$, $t_k, v_k \leftarrow \mathbb{Z}_N$, sets $\mathsf{ct}_k = (m_{b_k} \cdot e(g_1^{t_k}, h_1), g_1^{t_k} \cdot w^{v_k})$ and if $D(\mathsf{ct}_k) = b_k$, it sets $\mathsf{count} = \mathsf{count} + 1$, else sets $\mathsf{count} = \mathsf{count} - 1$. If $\mathsf{count} < \lambda/8$, the reduction algorithm outputs a uniformly random bit.

Otherwise, it uses D to break Assumption 4. It chooses $b \leftarrow \{0,1\}$, computes $\mathsf{ct} = (m_b \cdot T, h_3)$. If $D(\mathsf{ct}) = b$, it guesses that $T = e(g_1, g_1)^{\alpha \cdot s}$. Else it guesses that T is uniformly random.

First, note that the keys are distributed as output of $\mathsf{KeyGen}^{\mathsf{less}}$ algorithm. This argument follows from the Chinese Remainder Theorem, since

$$\{(g_1^{\alpha} \cdot w \cdot h_2^{u_j})_j : \gamma, \gamma', \delta, \delta', u_j \leftarrow \mathbb{Z}_N, w = g_3^{\gamma} \cdot g_4^{\delta}, h_2 = g_3^{\gamma'} \cdot g_4^{\delta'}\} \equiv \{(g_1^{\alpha} \cdot g_3^{\gamma_j} \cdot g_4^{\delta_j})_j : \gamma_j, \delta_j \leftarrow \mathbb{Z}_N\}$$

If T is a uniformly random element in $\mathbb{G}_T$, then D cannot distinguish between $m_0 \cdot T$ and $m_1 \cdot T$. Therefore, $\Pr[\mathcal{B}$ guesses $T = e(g_1, g_1)^{\alpha \cdot s} \mid T$ is random $] = 1/2$. Next, we will analyse the probability $\mathcal{B}$'s guess is correct if $T = e(g_1, g_1)^{\alpha \cdot s}$. Let event $\mathsf{Box}_{\delta}^{\mathcal{A}}$ denote the event that $\mathcal{A}$ outputs a δ-$\mathsf{Dist}^{\mathsf{less}}$ box D. Recall $\kappa = \epsilon - \epsilon \cdot (n+1)/2n$.

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s}] \\
&\quad = \Pr[\mathcal{B} \text{ guesses correctly} \wedge \mathsf{Box}_{\kappa}^{\mathcal{A}} \mid T = e(g_1, g_1)^{\alpha \cdot s}] \\
&\quad + \Pr[\mathcal{B} \text{ guesses correctly} \wedge \mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \wedge \neg\mathsf{Box}_{\kappa}^{\mathcal{A}} \mid T = e(g_1, g_1)^{\alpha \cdot s}]. \\
&\quad + \Pr[\mathcal{B} \text{ guesses correctly} \wedge \neg\mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \mid T = e(g_1, g_1)^{\alpha \cdot s}].
\end{aligned}$$

First, let us analyse the probability of $\mathcal{B}$ correctly guessing when $\mathcal{A}$ outputs a κ-$\mathsf{Dist}^{\mathsf{less}}$ box.

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\kappa}^{\mathcal{A}}] \\
&\quad = \Pr[\mathsf{count} \geq \lambda/8 \wedge D(\mathsf{ct}) = b \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\kappa}^{\mathcal{A}}] \\
&\quad + \frac{1}{2} \Pr[\mathsf{count} < \lambda/8 \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\kappa}^{\mathcal{A}}].
\end{aligned}$$

Using Chernoff bounds, we have that

$$\Pr\left[\mathsf{count} < \lambda/8 \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\kappa}^{\mathcal{A}}\right] = \mathrm{negl}(\lambda).$$

Also, we know that $\Pr\left[D(\mathsf{ct}) = b \mid \mathsf{Box}_{\kappa}^{\mathcal{A}}\right] \geq \frac{1}{2} + \kappa$. Thus, we can conclude that

$$\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\kappa}^{\mathcal{A}}] \geq \frac{1}{2} + \kappa - \mathrm{negl}(\lambda). \tag{15}$$

Next, let us analyse the probability of $\mathcal{B}$ correctly guessing when $\mathcal{A}$ outputs an $\epsilon/16$-$\mathsf{Dist}^{\mathsf{less}}$ box.

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \wedge \neg\mathsf{Box}_{\kappa}^{\mathcal{A}}] \\
&\quad = \Pr[\mathsf{count} \geq \lambda/8 \wedge D(\mathsf{ct}) = b \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \wedge \neg\mathsf{Box}_{\kappa}^{\mathcal{A}}] \\
&\quad + \frac{1}{2} \Pr[\mathsf{count} < \lambda/8 \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \wedge \neg\mathsf{Box}_{\kappa}^{\mathcal{A}}].
\end{aligned}$$

Let x be the probability that $\mathsf{count} \geq \lambda/8$ when $\mathcal{A}$ outputs such a box. Concretely, let $x = \Pr[\mathsf{count} < \lambda/8 | T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}^{\mathcal{A}}_{\epsilon/16} \wedge \neg\mathsf{Box}^{\mathcal{A}}_{\kappa}]$. Given this we can write that

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}^{\mathcal{A}}_{\epsilon/16} \wedge \neg\mathsf{Box}^{\mathcal{A}}_{\kappa}] \\
&\quad \geq x \cdot \left(\frac{1}{2} + \frac{\epsilon}{16}\right) + (1 - x) \cdot \frac{1}{2} \geq \frac{1}{2}.
\end{aligned} \tag{16}$$

Next, let us analyse the probability of $\mathcal{B}$ correctly guessing when $\mathcal{A}$ *does not* output an $\epsilon/16$-$\mathsf{Dist}^{\mathsf{less}}$ box.

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \neg\mathsf{Box}^{\mathcal{A}}_{\epsilon/16}] \\
&\quad = \Pr[\mathsf{count} \geq \lambda/8 \wedge D(\mathsf{ct}) = b \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \neg\mathsf{Box}^{\mathcal{A}}_{\epsilon/16}] \\
&\quad + \frac{1}{2} \Pr[\mathsf{count} < \lambda/8 \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \neg\mathsf{Box}^{\mathcal{A}}_{\epsilon/16}].
\end{aligned}$$

Again, using Chernoff bounds, we have that

$$\Pr\left[\mathsf{count} \geq \lambda/8 \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \neg\mathsf{Box}^{\mathcal{A}}_{\epsilon/16}\right] = \mathrm{negl}(\lambda).$$

Thus, we can conclude that

$$\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \neg\mathsf{Box}^{\mathcal{A}}_{\epsilon/16}] \geq \frac{1}{2} - \mathrm{negl}(\lambda). \tag{17}$$

Finally, we also have that

$$\Pr\left[\mathsf{Box}^{\mathcal{A}}_{\kappa} \mid T = e(g_1, g_1)^{\alpha \cdot s}\right] = \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\mathcal{A},\kappa}(\lambda, n+1) = \rho.$$

Combining above fact with Equations 15, 16 and 17, we get that

$$\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s}] \geq \frac{1}{2} + \rho \cdot \kappa - \mathrm{negl}(\lambda).$$

As a result, the advantage of $\mathcal{B}$ in breaking Assumption 4 is at least $\rho \cdot \kappa - \mathrm{negl}(\lambda)$. This completes the proof. ∎

From the above lemmas, it follows that $\Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\epsilon-\epsilon/2n}(\lambda, 1) - \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\epsilon-\epsilon(n+1)/2n}(\lambda, n+1) \geq \Pr\text{-}\mathsf{G}\text{-}\mathsf{D}_{\mathcal{A},n,\epsilon}(\lambda) - \mathrm{negl}(\lambda)$. This brings us to the following observation.

Observation C.2. For any PPT adversary $\mathcal{A}$, non-negligible function $\epsilon(\cdot)$, polynomials $n(\cdot), p(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, there exists a subset $\Gamma \subseteq \{1, 2, \ldots, n\}$ such that $\Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot i/2n}(\lambda, i) - \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot(i+1)/2n}(\lambda, i+1) > 0$ and

$$\sum_{i \in \Gamma} \left(\Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot i/2n}(\lambda, i) - \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot(i+1)/2n}(\lambda, i+1)\right) \geq \Pr\text{-}\mathsf{G}\text{-}\mathsf{D}_{\mathcal{A},n,\epsilon}(\lambda) - \mathrm{negl}(\lambda).$$

The next lemma will prove that $\Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}(i+1)$ and $\Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{leq}}(i)$ are approximately equal.

Lemma C.5. For any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $i \in \{1, 2, \ldots, n\}$, $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$,

$$\Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{leq}}_{\epsilon-\frac{\epsilon\cdot(i+1)}{2n}+\frac{\epsilon}{4n}}(\lambda, i) \leq \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\epsilon-\frac{\epsilon\cdot(i+1)}{2n}}(\lambda, i+1) + \mathrm{negl}(\lambda).$$

Proof. Let $\rho_1(\lambda) = \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{leq}}_{\epsilon-\frac{\epsilon\cdot(i+1)}{2n}+\frac{\epsilon}{4n}}(\lambda, i)$ and $\rho_2(\lambda) = \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\epsilon-\frac{\epsilon\cdot(i+1)}{2n}}(\lambda, i+1)$.

Let Expt_1 denote the first scenario, and Expt_2 the second one. The only differences in the two scenarios are as follows:

Key for user i in the first scenario is generated using $\mathsf{KeyGen}^{\mathsf{eq}}$, while in the second scenario, it is generated using $\mathsf{KeyGen}^{\mathsf{less}}$.

Key for user $i+1$ in the first scenario is generated using $\mathsf{KeyGen}^{\mathsf{gr}}$, while in the second scenario, it is generated using $\mathsf{KeyGen}^{\mathsf{eq}}$.

The decoder in the first scenario must distinguish between $\mathsf{Enc}^{\mathsf{leq}}$ encryptions with advantage at least $\epsilon - \frac{\epsilon\cdot(i+1)}{2n} + \frac{\epsilon}{4n}$, while the decoder in the second scenario must distinguish between encryptions generated using $\mathsf{Enc}^{\mathsf{less}}$ with advantage at least $\epsilon - \frac{\epsilon\cdot(i+1)}{2n}$.

We will construct two hybrid experiments, and show that consecutive hybrid experiments are computationally indistinguishable.

Hybrid Hyb_1: This is identical to Expt_1, except that the key for user i is generated using $\mathsf{KeyGen}^{\mathsf{less}}$. The key for user $i+1$ is generated using $\mathsf{KeyGen}^{\mathsf{gr}}$ and the decoder must distinguish between $\mathsf{Enc}^{\mathsf{leq}}$ encryptions with advantage at least $\gamma_1 = \epsilon - \frac{\epsilon\cdot(i+1)}{2n} + \frac{\epsilon}{8n}$. Let $\rho_{\mathsf{Hyb}_1}(\lambda)$ denote the probability that the decoder output can distinguish between $\mathsf{Enc}^{\mathsf{leq}}$ encryptions with advantage at least γ_1.

Hybrid Hyb_2 This is identical to Hyb_1, except that decoder must distinguish between $\mathsf{Enc}^{\mathsf{less}}$ encryptions with advantage at least $\gamma_2 = \epsilon - \frac{\epsilon\cdot(i+1)}{2n} + \frac{\epsilon}{16n}$. Let $\rho_{\mathsf{Hyb}_2}(\lambda)$ denote the probability that the decoder output can distinguish between $\mathsf{Enc}^{\mathsf{less}}$ encryptions with advantage at least γ_2.

Claim C.5. Assuming the subgroup decision assumption (Assumption 3), for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}_1(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_1(\lambda) - \rho_{\mathsf{Hyb}_1}(\lambda) \leq \text{negl}_1(\lambda)$.

Proof. Suppose $\rho_1 - \rho_{\mathsf{Hyb}_1} = \eta$ for some non-negligible function η. We will construct a reduction algorithm $\mathcal{B}$ that breaks the subgroup decision assumption with non-negligible advantage.

First, $\mathcal{B}$ sends its challenge sets $S_0 = \{2,4\}$, $S_1 = \{3,4\}$ and it receives T, where $T \in \mathbb{G}_{S_0}$ or $T \in \mathbb{G}_{S_1}$. Next, it queries for the generators for $\mathbb{G}_{\{1\}}$, $\mathbb{G}_{\{2,3\}}$, $\mathbb{G}_{3,4}$ and $\mathbb{G}_{\{4\}}$, and receives $g_1, g_{2,3}$, $g_{3,4}$ and g_4 respectively. The reduction algorithm first chooses $\alpha \leftarrow \mathbb{Z}_N$ and sets $\mathsf{mpk} = (e(g_1,g_1)^\alpha, g_1)$. Next, it uses g_1 and $g_{3,4}$ to construct keys for indices less than i; that is, it chooses $u_j \leftarrow \mathbb{Z}_N$ and sets $\mathsf{sk}_j = g_1^\alpha \cdot g_{3,4}^{u_j}$ as the secret key for index $j < i$. For the i^{th} index, it uses g_1 and T; that is, it sets $\mathsf{sk}_i = g_1^\alpha \cdot T$. Finally, for indices $j > i$, the reduction algorithm uses g_1 and g_4 and sets $\mathsf{sk}_j = g_1^\alpha \cdot g_4^{u_j}$ for randomly chosen $u_j \leftarrow \mathbb{Z}_N$.

After all secret key queries, the reduction algorithm receives pirate box D and messages m_0, m_1. The reduction algorithm sets $\gamma = \epsilon - \epsilon\cdot(i+1)/2n + \epsilon/6n$, $z = \lambda \cdot n/\epsilon$ and tests whether D is a γ-$\mathsf{Dist}^{\mathsf{less}}$ box for m_0, m_1. The reduction algorithm first sets $\mathsf{count} = 0$. For $k = 1$ to z, it chooses $b_k \leftarrow \{0,1\}$, $s_k, t_k \leftarrow \mathbb{Z}_N$, sets $\mathsf{ct}_k = (m_{b_k} \cdot e(g_1,g_1)^{\alpha\cdot s_k}, g_1^{s_k} \cdot g_{2,3}^{t_k})$ and if $D(\mathsf{ct}_k) = b_k$, it sets $\mathsf{count} = \mathsf{count} + 1$, else it sets $\mathsf{count} = \mathsf{count} - 1$. Finally, after the z iterations, if $\mathsf{count} > \gamma \cdot z$, then $\mathcal{B}$ guesses that $T \in \mathbb{G}_{S_0}$, else it guesses that $T \in \mathbb{G}_{S_1}$.

Let us now compute the reduction algorithm's advantage. First, note that if $T \leftarrow \mathbb{G}_{S_0}$ then the key for index i corresponds to a $\mathsf{KeyGen}^{\mathsf{eq}}$ key, and if $T \in \mathbb{G}_{S_1}$, then it corresponds to a $\mathsf{KeyGen}^{\mathsf{less}}$ key. For indices $j < i$, the adversary gets $\mathsf{KeyGen}^{\mathsf{less}}$ keys (we use the Chinese Remainder Theorem to argue that $\{(g_1^\alpha \cdot g_{3,4}^{u_j})_j \ : \ g_1 \leftarrow \mathbb{G}_1, g_{3,4} \leftarrow \mathbb{G}_{\{3,4\}}, u_j \leftarrow \mathbb{Z}_N\}$ is statistically indistinguishable from $\{(g_1^\alpha \cdot g_3^{u_j} \cdot g_4^{v_j}) \ : \ g_1 \leftarrow \mathbb{G}_{\{1\}}, g_3 \leftarrow \mathbb{G}_{\{3\}}, g_4 \leftarrow \mathbb{G}_{\{4\}}\}$). Similarly, for all indices $j > i$, the keys are generated using $\mathsf{KeyGen}^{\mathsf{gr}}$.

Similarly, using Chinese Remainder Theorem, we can argue that the ciphertexts computed by the reduction algorithm are indistinguishable from $\mathsf{Enc}^{\mathsf{leq}}$ ciphertexts. The analysis of $\mathcal{B}$'s advantage is similar to that in proof of Claim C.1.

■

Claim C.6. Assuming the subgroup decision assumption (Assumption 3), for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$,there exists a negligible function $\text{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_{\mathsf{Hyb}_1}(\lambda) - \rho_{\mathsf{Hyb}_2}(\lambda) \leq \text{negl}_2(\lambda)$.

Proof. Suppose $\rho_{\mathsf{Hyb}_1} - \rho_{\mathsf{Hyb}_2} = \eta$ for some non-negligible function η. We will construct a reduction algorithm $\mathcal{B}$ that breaks the subgroup decision assumption with non-negligible advantage.

First, $\mathcal{B}$ sends its challenge sets $S_0 = \{2, 3\}$, $S_1 = \{3\}$ and it receives T, where $T \in \mathbb{G}_{S_0}$ or $T \in \mathbb{G}_{S_1}$. Next, it queries for the generators for $\mathbb{G}_{\{1\}}$, $\mathbb{G}_{\{3\}}$, $\mathbb{G}_{\{4\}}$, and receives g_1, g_3, g_4 respectively. First, it chooses $\alpha \leftarrow \mathbb{Z}_N$ and sends $\mathsf{mpk} = (e(g_1, g_1)^\alpha, g_1)$ to $\mathcal{A}$. Next, it receives key queries from $\mathcal{A}$, and it uses g_1, g_3 and g_4 to construct keys. For indices $j \le i$, it chooses $u_j, t_j \leftarrow \mathbb{Z}_N$ and sets $\mathsf{sk}_j = g_1^\alpha \cdot g_3^{u_j} \cdot g_4^{t_j}$. For indices $j > i$, the reduction algorithm chooses $t_j \leftarrow \mathbb{Z}_N$ and sends $\mathsf{sk}_j = g_1^\alpha \cdot g_4^{t_j}$. Finally, after all secret key queries, the reduction algorithm receives pirate box D and m_0, m_1. It sets $\gamma = \epsilon - \epsilon \cdot (i+1)/2n + \epsilon/12n$, $z = \lambda \cdot n/\epsilon$ and $\mathsf{count} = 0$.

For $k = 1$ to z, the reduction algorithm chooses $b_k \leftarrow \{0,1\}$, $s_k, t_k \leftarrow \mathbb{Z}_N$, sets $\mathsf{ct}_k = (m_{b_k} \cdot e(g_1, g_1)^{\alpha \cdot s_k}, g_1^{s_k} \cdot T^{t_k})$ and checks if $D(\mathsf{ct}_k) = b_k$. If so, it sets $\mathsf{count} = \mathsf{count} + 1$, else it sets $\mathsf{count} = \mathsf{count} - 1$. After the z iterations, $\mathcal{B}$ checks if $\mathsf{count} > \gamma \cdot z$. If so, then $\mathcal{B}$ guesses that $T \in \mathbb{G}_{S_0}$, else it guesses that $T \in \mathbb{G}_{S_1}$.

First, note that the secret keys sent to $\mathcal{A}$ are identically distributed as in Hyb_1 and Hyb_2 experiments. Using the Chinese Remainder Theorem, we can argue that if $T \leftarrow \mathbb{G}_{S_0}$, then the z ciphertexts constructed are distributed as z encryptions generated using $\mathsf{Enc}^{\mathsf{leq}}$; if $T \leftarrow \mathbb{G}_{S_1}$, then the z ciphertexts are distributed as z encryptions using $\mathsf{Enc}^{\mathsf{less}}$. The analysis of $\mathcal{B}$'s advantage is similar to that in proof of Claim C.1. ∎

Claim C.7. Assuming the subgroup decision assumption (Assumption 3), for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}_3(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_{\mathsf{Hyb}_2}(\lambda) - \rho_2(\lambda) \le \text{negl}_3(\lambda)$.

Proof. Suppose $\rho_{\mathsf{Hyb}_2} - \rho_2 = \eta$ for some non-negligible function η. We will construct a reduction algorithm $\mathcal{B}$ that breaks the subgroup decision assumption with non-negligible advantage.

First, $\mathcal{B}$ sends its challenge sets $S_0 = \{4\}$, $S_1 = \{2, 4\}$ and it receives T, where $T \in \mathbb{G}_{S_0}$ or $T \in \mathbb{G}_{S_1}$. Next, it queries for the generators for $\mathbb{G}_{\{1\}}$, $\mathbb{G}_{\{3\}}$, $\mathbb{G}_{\{4\}}$, and receives g_1, g_3, g_4 respectively.

The reduction algorithm first chooses $\alpha \leftarrow \mathbb{Z}_N$ and sets $\mathsf{mpk} = (e(g_1, g_1)^\alpha, g_1)$. Next, it uses g_1 and g_3 and g_4 to construct keys for indices less than or equal to i; that is, it chooses $u_j, t_j \leftarrow \mathbb{Z}_N$ and sets $\mathsf{sk}_j = g_1^\alpha \cdot g_3^{u_j} \cdot g_4^{t_j}$ as the secret key for index $j \le i$. For the $(i+1)^{th}$ index, it uses g_1 and T; that is, it sets $\mathsf{sk}_{i+1} = g_1^\alpha \cdot T$. Finally, for indices $j > i+1$, the reduction algorithm uses g_1 and g_4 and sets $\mathsf{sk}_j = g_1^\alpha \cdot g_4^{u_j}$ for randomly chosen $u_j \leftarrow \mathbb{Z}_N$.

After all secret key queries, the reduction algorithm receives pirate box D and m_0, m_1. The reduction algorithm sets $\gamma = \epsilon - \epsilon \cdot (i+1)/2n + \epsilon/32n$, $z = \lambda \cdot n/\epsilon$ and tests whether D is a γ-$\mathsf{Dist}^{\mathsf{less}}$ box for m_0, m_1. The reduction algorithm first sets $\mathsf{count} = 0$. For $k = 1$ to z, it chooses $b_k \leftarrow \{0,1\}$, $s_k, t_k \leftarrow \mathbb{Z}_N$, sets $\mathsf{ct}_k = (m_{b_k} \cdot e(g_1, g_1)^{\alpha \cdot s_k}, g_1^{s_k} \cdot g_3^{t_k})$ and if $D(\mathsf{ct}_k) = b_k$, it sets $\mathsf{count} = \mathsf{count} + 1$, else it sets $\mathsf{count} = \mathsf{count} - 1$. Finally, after the z iterations, if $\mathsf{count} > \gamma \cdot z$, then $\mathcal{B}$ guesses that $T \in \mathbb{G}_{S_0}$, else it guesses that $T \in \mathbb{G}_{S_1}$. The analysis of $\mathcal{B}$'s advantage is similar to that in proof of Claim C.1. ∎

∎

Lemma C.6. For any PPT adversary $\mathcal{A}$, polynomial $n(\cdot)$ and non-negligible function $\epsilon(\cdot)$, all $\lambda \in \mathbb{N}$,

$$\Pr\text{-}\mathsf{Gap}_{\epsilon/4n}(\lambda, i) \ge \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\epsilon - \epsilon \cdot i/2n}(\lambda, i) - \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{leq}}_{\epsilon - \epsilon \cdot i/2n - \epsilon/4n}(\lambda, i).$$

Proof. Recall that $\Pr\text{-}\mathsf{Gap}$ is defined as below

$$\Pr\text{-}\mathsf{Gap}_{\mathcal{A},n,\epsilon/4n}(\lambda, i) = \Pr\left[\exists\ \delta \in [0, 1/2] \text{ s.t. } \begin{array}{l} D \text{ is } \delta\text{-}\mathsf{Dist}^{\mathsf{less}} \wedge \\ D \text{ is not } (\delta - \frac{\epsilon}{4n})\text{-}\mathsf{Dist}^{\mathsf{leq}} \end{array} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)\right].$$

Now we can also write that

$$\Pr\text{-}\mathsf{Gap}_{\mathcal{A},n,\epsilon/4n}(\lambda, i) \ge \max_{\delta \in [0,1/2]} \Pr\left[\begin{array}{l} D \text{ is } \delta\text{-}\mathsf{Dist}^{\mathsf{less}} \wedge \\ D \text{ is not } (\delta - \frac{\epsilon}{4n})\text{-}\mathsf{Dist}^{\mathsf{leq}} \end{array} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)\right].$$

We also know that for any $\delta \in [0, 1/2]$,

$$\begin{aligned}
&\Pr\left[\begin{array}{l} D \text{ is } \delta\text{-}\mathsf{Dist}^{\mathsf{less}} \wedge \\ D \text{ is not } (\delta - \frac{\epsilon}{4n})\text{-}\mathsf{Dist}^{\mathsf{leq}} \end{array} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)\right] \\
&\geq \Pr\left[D \text{ is } \delta\text{-}\mathsf{Dist}^{\mathsf{less}} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)\right] \\
&\quad - \Pr\left[D \text{ is } (\delta - \frac{\epsilon}{4n})\text{-}\mathsf{Dist}^{\mathsf{leq}} : (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)\right] \\
&\geq \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\delta}(\lambda, i) - \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{leq}}_{\delta - \epsilon/4n}(\lambda, i).
\end{aligned}$$

Finally substituting $\delta = \epsilon - \epsilon \cdot i/2n$, we get

$$\Pr\text{-}\mathsf{Gap}_{\epsilon/4n}(\lambda, i) \geq \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\epsilon - \epsilon \cdot i/2n}(\lambda, i) - \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{leq}}_{\epsilon - \epsilon \cdot i/2n - \epsilon/4n}(\lambda, i).$$

This concludes the proof. ■

C.2.5 Proof of Theorem C.4

We need to show that for any PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$, non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\text{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$,

$$\Pr\text{-}\mathsf{Cor}\text{-}\mathsf{Tr}_{\mathcal{A},n,\epsilon}(\lambda) \geq \frac{\sum_i \Pr\text{-}\mathsf{Gap}_{\mathcal{A},n,\epsilon/4n}(\lambda, i)}{n(\lambda)} - \text{negl}(\lambda).$$

First, consider the following events $\mathsf{Tr}_{\mathcal{A},n,\epsilon}$ and $\mathsf{Tr}'_{\mathcal{A},n,\epsilon}$, parameterized by $\mathcal{A}, n, \epsilon$. The event Tr is similar to $\mathsf{Cor}\text{-}\mathsf{Tr}$, except that the output of the trace should be in $\{1, 2, \ldots, n\}$ (in particular, it is not required that the output be in the set S of keys queried). Let $\rho_{\mathsf{less}} = \Pr[b \leftarrow D(\mathsf{ct}) : b \leftarrow \{0, 1\}, \mathsf{ct} \leftarrow \mathsf{Enc}^{\mathsf{less}}(\mathsf{msk}, m_b)]$ and $\rho_{\mathsf{leq}} = \Pr[b \leftarrow D(\mathsf{ct}) : b \leftarrow \{0, 1\}, \mathsf{ct} \leftarrow \mathsf{Enc}^{\mathsf{leq}}(\mathsf{msk}, m_b)]$.

The event Tr' is defined similar to Tr, except we say that Tr' occurs if D is ϵ-Dist box and $\rho_{\mathsf{less}} - \rho_{\mathsf{leq}} > \epsilon/4n$. Note that the only difference between Tr and Tr' is that in Tr, the challenger computes an estimates $\hat{\rho}_{\mathsf{less}}$ and $\hat{\rho}_{\mathsf{leq}}$ of $\rho_{\mathsf{less}}, \rho_{\mathsf{leq}}$ (respectively), and checks if $\hat{\rho}_{\mathsf{less}} - \hat{\rho}_{\mathsf{leq}} > \epsilon/8n$.

Now, using Chernoff bounds, it follows that $\Pr[\mathsf{Tr}_{\mathcal{A},n,\epsilon}] \geq \Pr[\mathsf{Tr}'_{\mathcal{A},n,\epsilon}] - 2^{-O(\lambda)}$. Next, it follows from the definitions of $\Pr[\mathsf{Tr}'_{\mathcal{A},n,\epsilon}]$ and $\Pr\text{-}\mathsf{Gap}_{\mathcal{A},n,\gamma}$ that $\Pr[\mathsf{Tr}'_{\mathcal{A},n,\epsilon}] = \sum_i \Pr\text{-}\mathsf{Gap}_{\mathcal{A},n,\epsilon/4n}(\lambda, i)/n$.

Finally, note that

$$\begin{aligned}
\Pr[\mathsf{Cor}\text{-}\mathsf{Tr}] &= \Pr[\mathsf{Tr}] - \Pr[\mathsf{Tr} \wedge \text{Trace outputs } i \notin S] \\
&= \Pr[\mathsf{Tr}] - \Pr[\mathsf{Fal}\text{-}\mathsf{Tr}] \\
&\geq \Pr[\mathsf{Tr}] - \text{negl}_1(\lambda) \text{ (using Theorem C.2)} \\
&\geq \Pr[\mathsf{Tr}'] - 2^{-O(\lambda)} - \text{negl}_1(\lambda) \\
&\geq \sum_i \frac{\Pr\text{-}\mathsf{Gap}_{\mathcal{A},n,\epsilon/4n}(\lambda, i)}{n} - \text{negl}(\lambda).
\end{aligned}$$

This concludes the proof.