# Risky Traitor Tracing and New Differential Privacy Negative Results

Rishab Goyal* Venkata Koppula† Brent Waters‡

November 21, 2017

**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 composite order bilinear groups with assumptions close to those used in prior works. Our core system achieves $f(\lambda, n) = \frac{1}{n}$ where ciphertexts consists of three group elements and decryption requires just two pairing operations. In addition, we show a generic way to increase $f$ by approximately a factor of $k$ if we increase the size of the ciphertext and decryption time also by a factor of $k$.

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 three 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. One potential use of a risky traitor tracing system is to place it in a continual use situation where users will periodically receive fresh traitor tracing secret keys via a key refresh mechanism that is built using standard encryption. If an attacker wishes to produce a decoder $D$ that continues to work through these refreshes the decoder will be taking an $f$ risk of being caught after *each* key refresh, which presents a Russian roulette situation for the attacker.
3. Finally, we can capture impossibility results for differential privacy from risky traitor tracing. Since our ciphertexts are short ($O(\lambda)$), thus 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 Introduction

A traitor tracing [CFN94] system is an encryption system in which a setup algorithm produces a public key $\mathsf{pk}$, master secret key $\mathsf{msk}$ 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 $\mathsf{ct}$ 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

*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: bwaters@cs.utexas.edu. Supported by NSF CNS-1228599 and CNS-1414082, DARPA SafeWare, Microsoft Faculty Fellowship, and Packard Foundation Fellowship.

produces a decryption algorithm $D$ that decrypts ciphertext with some non-negligible probability $\epsilon(\lambda)$ where $\lambda$ is the security parameter. An additional Trace algorithm will take as input the master secret key $\mathsf{msk}$ 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 $\lambda$ and the number of users $n$. Like a traitor tracing system it output a public key $\mathsf{pk}$, master secret key $\mathsf{msk}$ and $n$ private keys $\mathsf{sk}_1, \mathsf{sk}_2, \ldots, \mathsf{sk}_n$ where a user with index $j$ is given key $\mathsf{sk}_j$. Any of the private keys is capable of decrypting a ciphertext $\mathsf{ct}$ created using $\mathsf{pk}$. 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 composite order bilinear groups with assumptions close to those used in [BSW06]. Our core system achieves $f(\lambda, n) = \frac{1}{n}$ where ciphertexts consist of three group elements and decryption requires just two pairing operations. In addition, we show a generic way to increase $f$ by approximately a factor of $k$ at the cost of increasing the size of the ciphertext and decryption time also by a factor of $k$.

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 three 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. One potential use of a risky traitor tracing system is to place it in a continual use situation where users will periodically receive fresh traitor tracing private keys via a key refresh mechanism that is built using standard encryption. If an attacker wishes to produce a decoder $D$ that continues to work through these refreshes the decoder will be taking an $f$ risk of being caught after *each* key refresh, which present a Russian roulette situation for the attacker. In this scenario 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. Finally, we show that the argument of Dwork et al. applies to 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 ($O(\lambda)$), thus 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 $\lambda$, and number of users $n$ and generates

a public key $\mathsf{pk}$, a master secret key $\mathsf{msk}$, and $n$ private keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_n$. The encrypt algorithm encrypts messages using $\mathsf{pk}$ and the decrypt algorithm decrypts a ciphertext using any one of the private keys $\mathsf{sk}_i$. The tracing algorithm takes $\mathsf{msk}$ 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 $\mathsf{sk}_i$ was used to create the pirate decoder.

Traditionally, a traitor tracing scheme is required to satisfy two security properties. First, it must be IND-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.[1] 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 $\lambda$ and number of users $n$ as inputs. The syntax as well as IND-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 uses composite order bilinear groups. 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.

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

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 $\alpha$, 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 $\rho$, then we can identify a traitor with probability $\rho/n$. To prove this, we show that if $\rho_i$ denotes the probability that the adversary outputs a $\rho$-functional box and $i$ is the cutoff-index, then the sum of all these $\rho_i$ quantities is close to $\rho$. More details are provided in Section 4.2.

**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 only support encryptions to two adjacent indices (that is, if $i$ is the cutoff index, then we only support encryptions to either $i$ or $i - 1$) and index 0. As a result, the trace algorithm can only trace index $i$. Our proof uses composite order assumptions similar to those used in [BSW06]. The main difficulty in our proof argument is that encrypting to index $j$ is not defined for indices $j \notin \{0, i-1, i\}$. As a result, we need to come up with a new way to link success probabilities across different setups and weave 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 $\mathsf{ct}$, the corresponding query $q_{\mathsf{ct}}$ on input a private key $\mathsf{sk}$ outputs the decryption of $\mathsf{ct}$ using $\mathsf{sk}$. 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, it 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 $s$.[2]

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.[3] 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 $\delta$, 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 reject symbol. 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 Section 5.2.

**Amplifying the Probability of Tracing — Using Risky Traitor Tracing in a Continuous Setting.** 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. 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 $\mathsf{pk}$ to compute

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

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

a ciphertext. A user $i$ will use secret key $\mathsf{sk}_i$ 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 $\mathsf{pk}'$ 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 $\mathsf{sk}'_i$ under the respective permanent secret key $\mathrm{SK}_i$. And the encryptors will be instructed to only encrypt using the new public key $\mathsf{pk}'$. 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 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 $\delta$ then the secret key grows to $O(n^2\lambda/\delta^2)$. Furthermore, the system must be configured a-priori with a specific $\delta$ 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 $\mathsf{poly}(n)$.[4] [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.

# 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 $D(x)$).

## 2.1 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 $\lambda$ 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 1** (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 1.

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 $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\mathsf{Adv}_{\mathcal{A}}(\lambda) = \left|\Pr\left[1 \leftarrow \mathsf{Expt\text{-}GSD}_{\mathsf{Bilin\text{-}Gen}_k, \mathcal{A}}(\lambda)\right] - \frac{1}{2}\right| \leq \mathrm{negl}(\lambda)$.

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 2** (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 2.

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

**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 1: Experiment Expt-GSD

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

1. Challenger chooses $((p_1,\ldots,p_k),N,\mathbb{G},\mathbb{G}_T,e(\cdot,\cdot)) \leftarrow \mathsf{Bilin\text{-}Gen}_k(1^\lambda)$. 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 $S_1,S_2 \subseteq \{2,\ldots,k\}$ 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 2: Experiment Expt-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 3.** 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 $\lambda \in \mathbb{N}$,

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

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

# 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 $\mathsf{Trace}$ 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 $\lambda$, number of users $n$, and outputs a master secret key $\mathsf{msk}$, a public key $\mathsf{pk}$ 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 $\mathsf{pk}$, message $m \in \mathcal{M}$ and outputs a ciphertext $\mathsf{ct}$.

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

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

**Correctness** For correctness, we require that if $\mathsf{ct}$ is an encryption of message $m$, then decryption of $\mathsf{ct}$ 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 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 $\lambda \in \mathbb{N}$, $|\Pr[1 \leftarrow \mathsf{Expt\text{-}IND\text{-}CPA}^{\mathcal{T}}_{\mathcal{A}}(1^\lambda, 1^n)] - 1/2| \leq \mathrm{negl}(\lambda)$, 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 $\mathsf{Expt}^{\mathcal{T}}_{\mathcal{A},n,\epsilon}(\lambda)$:

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

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 $\mathsf{sk}_i$. Let $\mathcal{S}$ 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 $\lambda$, parameterized by $\mathcal{A}, n, \epsilon$):

- $\mathsf{Good\text{-}Decoder}$ : $\Pr[D(\mathsf{ct}) = m \mid m \leftarrow \mathcal{M}, \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m)] \geq \epsilon(\lambda)$
  $\Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Good\text{-}Decoder}]$.
- $\mathsf{Cor\text{-}Tr}$ : $i^* \in \mathcal{S}$
  $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Cor\text{-}Tr}]$.
- $\mathsf{Fal\text{-}Tr}$ : $i^* \in \{1, 2, \ldots, n\} \setminus \mathcal{S}$
  $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Fal\text{-}Tr}]$.

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 $\mathsf{Trace}$ 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 $\mathsf{msk}$, $y$ (in unary) and two messages $m_0, m_1$. It outputs $i \in \{1, 2, \ldots, n\} \cup \{\perp\}$.

**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 3. Based on this experiment, we will now define the following (probabilistic) events and the corresponding probabilities (which is a function of $\lambda$, parameterized by $\mathcal{A}, n, \epsilon$):

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

- $\mathsf{Fal\text{-}Tr}$ : $i^* \in \{1, 2, \ldots, n\} \setminus \mathcal{S}$
  $\Pr\text{-}\mathsf{Fal\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) = \Pr[\mathsf{Fal\text{-}Tr}]$.

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

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

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 $\mathsf{sk}_i$. Let $\mathcal{S}$ be the set of indices queried by $\mathcal{A}$.

Figure 3: 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 $\lambda$ and the number of users.

**Definition 3.4** ($f$-risky secure traitor tracing)**.** Let $f : \mathbb{N}\times\mathbb{N} \to [0, 1]$ 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 3). Based on this experiment, we will now define the following (probabilistic) events and the corresponding probabilities (which are functions of $\lambda$, parameterized by $\mathcal{A}, n, \epsilon$):

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

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 $\mathrm{negl}_1(\cdot)$, $\mathrm{negl}_2(\cdot)$ 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 $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) \cdot f(\lambda, n(\lambda)) - \mathrm{negl}_2(\lambda)$.

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 $m_0, m_1$,

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

where the probability is taken over random coins of $\mathsf{Trace}$.

## 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 $\lambda$, number of users $n$, and outputs a master secret key $\mathsf{msk}$ 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 $\mathsf{pk}$, message $m \in \mathcal{M}$ and outputs a ciphertext $\mathsf{ct}$.

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

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

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 4). Based on this experiment, we will now define the following (probabilistic) events and the corresponding probabilities (which is a function of $\lambda$, parameterized by $\mathcal{A}, n, \epsilon$):

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

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 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 $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) \cdot f(\lambda, n(\lambda)) - \mathrm{negl}_2(\lambda)$.

**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 4), 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 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 $\Pr\text{-}\mathsf{Cor\text{-}Tr}_{\mathcal{A},n,\epsilon}(\lambda) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) \cdot f(\lambda, n(\lambda)) - \mathrm{negl}_2(\lambda)$.

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

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

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 $\mathsf{sk}_i$. Let $S$ be the set of indices queried by $\mathcal{A}$.
The oracle $O_2(\cdot)$ is the encryption oracle that has $\mathsf{msk}$ hardwired, takes a message $m$ as input, and outputs $\mathsf{Enc}(\mathsf{msk}, m)$.

Figure 4: Experiment $\mathsf{Expt\text{-}TT\text{-}priv}$

# 4 Public Key Traitor Tracing Scheme

We will now present our public key traitor tracing scheme.

## 4.1 Construction

Let $\mathsf{Bilin\text{-}Gen}_4$ be a group generator that takes as input security parameter $\lambda$, 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 $\lambda$-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 $1 \le j \le 4$, $\alpha \leftarrow \mathbb{Z}_N$ and sets $\mathsf{mpk} = (e(g_1, g_1)^\alpha, g_1)$.

The key generation algorithm then chooses an index $i^* \leftarrow \{1, 2, \ldots, n\}$ 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 $\mathsf{msk}$), or outputs $\perp$.

## 4.2 Proof of Security

### 4.2.1 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 4.1.** Assuming the Bilinear Decisional Diffie Hellman assumption (Assumption 3), 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 $\epsilon$ 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 $\epsilon$.

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 $\epsilon$, then $\mathcal{B}$ breaks the BDDH assumption with advantage $\epsilon$.

■

### 4.2.2 False-Trace Probability

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

**Theorem 4.2.** 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 4.1.** 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 $X^{\mathsf{type}}$ 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 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 4.2.** Assuming the subgroup hiding assumption (Assumption 1), 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 $\eta$ 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 $\mathsf{mpk}$ using $g_1, \alpha$. 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 $\Pr[\mathcal{B} \text{ guesses } T \in \mathbb{G}_{S_1} \mid T \in \mathbb{G}_{S_1}]$.

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

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. ∎

∎

#### 4.2.3 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 $\gamma$-$\mathsf{Dist}$ 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 $\gamma$-$\mathsf{Dist}^{\mathsf{less}}$ 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 $\gamma$-$\mathsf{Dist}^{\mathsf{leq}}$ 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 5). The experiment takes as input a security parameter $\lambda$, 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 $\mathsf{mpk}$ 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 $\mathsf{sk}_j$.
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 5: Experiment $\mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)$

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

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

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

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

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

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 4.3.** 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 4.4.** 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.

### 4.2.4 Proof of Theorem 4.3

For notational simplicity, we will skip the dependence of $n$ and $\epsilon$ on $\lambda$. 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 4.1, Lemma 4.3 and Lemma 4.4).
- 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 4.2).
- 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 4.5).
- 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

$$\begin{aligned}\Pr\text{-}\mathsf{Gap}_{\epsilon/4n}(i) &\geq \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot i/2n}(i) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{leq}}_{\epsilon-\epsilon\cdot(i+1)/2n+\epsilon/4n}(i) \text{ (see Lemma 4.6)}\\ &\approx \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot i/2n}(i) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot(i+1)/2n}(i+1)\end{aligned}$$

- 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 4.1.** 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 4.3.** Assuming the subgroup decision assumption (Assumption 1), 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 $\lambda$, we will define a sequence of hybrid events, and show that the difference in their probabilities is at most negligible in $\lambda$.

**Hybrid $\mathsf{Hyb}_1$** 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 $\gamma_1$-$\mathsf{Dist}$ for $m_0, m_1$.

**Hybrid** $\mathsf{Hyb}_2$ 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 $\gamma_2$-$\mathsf{Dist}$ for $m_0, m_1$.

**Hybrid** $\mathsf{Hyb}_3$ This experiment is similar to that associated with $\mathsf{Hyb}_2$, 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 $\gamma_3$-$\mathsf{Dist}$ for $m_0, m_1$.

**Claim 4.1.** Assuming the subgroup decision assumption (Assumption 1), 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 $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_1(\lambda) - \rho_{\mathsf{Hyb}_1}(\lambda) \leq \mathrm{negl}_1(\lambda)$.

*Proof.* Suppose $\rho_1 - \rho_{\mathsf{Hyb}_1} = \eta$ for some non-negligible function $\eta$. 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 $\gamma$-$\mathsf{Dist}$ 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 4.1. 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

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

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 4.2.** Assuming the subgroup decision assumption (Assumption 1), 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 $\eta$. 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 4.1.

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 $\gamma$-$\mathsf{Dist}$ 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 4.1. ■

**Claim 4.3.** Assuming the subgroup decision assumption (Assumption 1), for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}_3(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_{\mathsf{Hyb}_2}(\lambda) - \rho_{\mathsf{Hyb}_3}(\lambda) \leq \mathrm{negl}_3(\lambda)$.

*Proof.* Suppose $\rho_{\mathsf{Hyb}_2} - \rho_{\mathsf{Hyb}_3} = \eta$ for some non-negligible function $\eta$. 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 4.1.

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 $\gamma$-$\mathsf{Dist}$ 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 4.1. ■

**Claim 4.4.** Assuming the subgroup decision assumption (Assumption 1), for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}_3(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_{\mathsf{Hyb}_3}(\lambda) - \rho_2(\lambda) \leq \mathrm{negl}_3(\lambda)$.

*Proof.* Suppose $\rho_{\mathsf{Hyb}_3} - \rho_2 = \eta$ for some non-negligible function $\eta$. 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 4.1.

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 $\gamma$-$\mathsf{Dist}$ 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 4.1. ■

■

**Lemma 4.4.** Assuming the subgroup hiding in target group assumption (Assumption 2), 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 $\epsilon(\lambda) > 1/p(\lambda)$,

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

*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 $\lambda$. 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 2 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 $\gamma$-$\mathsf{Dist}^{\mathsf{less}}$ box for $m_0, m_1$. Concretely, for $k = 1$ to $\lambda/\epsilon$, 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 2. 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 $\delta$-$\mathsf{Dist}^{\mathsf{less}}$ box $D$. Recall $\kappa = \epsilon - \epsilon \cdot (n+1)/2n$.

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s}] \\
&= \Pr[\mathcal{B} \text{ guesses correctly} \wedge \mathsf{Box}_{\kappa}^{\mathcal{A}} \mid T = e(g_1, g_1)^{\alpha \cdot s}] \\
&+ \Pr[\mathcal{B} \text{ guesses correctly} \wedge \mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \wedge \neg \mathsf{Box}_{\kappa}^{\mathcal{A}} \mid T = e(g_1, g_1)^{\alpha \cdot s}]. \\
&+ \Pr[\mathcal{B} \text{ guesses correctly} \wedge \neg \mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \mid T = e(g_1, g_1)^{\alpha \cdot s}].
\end{aligned}$$

First, let us analyse the probability of $\mathcal{B}$ correctly guessing when $\mathcal{A}$ outputs a $\kappa$-$\mathsf{Dist}^{\mathsf{less}}$ box.

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\kappa}^{\mathcal{A}}] \\
&= \Pr[\mathsf{count} \geq \lambda/8 \wedge D(\mathsf{ct}) = b \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\kappa}^{\mathcal{A}}] \\
&+ \frac{1}{2} \Pr[\mathsf{count} < \lambda/8 \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\kappa}^{\mathcal{A}}].
\end{aligned}$$

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

$$\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\kappa}^{\mathcal{A}}] \geq \frac{1}{2} + \kappa - \mathrm{negl}(\lambda). \tag{1}$$

Next, let us analyse the probability of $\mathcal{B}$ correctly guessing when $\mathcal{A}$ outputs an $\epsilon/16$-$\mathsf{Dist}^{\mathsf{less}}$ box.

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \wedge \neg \mathsf{Box}_{\kappa}^{\mathcal{A}}] \\
&= \Pr[\mathsf{count} \geq \lambda/8 \wedge D(\mathsf{ct}) = b \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \wedge \neg \mathsf{Box}_{\kappa}^{\mathcal{A}}] \\
&+ \frac{1}{2} \Pr[\mathsf{count} < \lambda/8 \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \wedge \neg \mathsf{Box}_{\kappa}^{\mathcal{A}}].
\end{aligned}$$

Let $x$ be the probability that $\mathsf{count} \geq \lambda/8$ when $\mathcal{A}$ outputs such a box. Concretely, let $x = \Pr[\mathsf{count} < \lambda/8 \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \wedge \neg \mathsf{Box}_{\kappa}^{\mathcal{A}}]$. Given this we can write that

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \mathsf{Box}_{\epsilon/16}^{\mathcal{A}} \wedge \neg \mathsf{Box}_{\kappa}^{\mathcal{A}}] \\
&\geq x \cdot \left(\frac{1}{2} + \frac{\epsilon}{16}\right) + (1 - x) \cdot \frac{1}{2} \geq \frac{1}{2}.
\end{aligned} \tag{2}$$

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.

$$\begin{aligned}
&\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \neg \mathsf{Box}_{\epsilon/16}^{\mathcal{A}}] \\
&= \Pr[\mathsf{count} \geq \lambda/8 \wedge D(\mathsf{ct}) = b \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \neg \mathsf{Box}_{\epsilon/16}^{\mathcal{A}}] \\
&+ \frac{1}{2} \Pr[\mathsf{count} < \lambda/8 \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \neg \mathsf{Box}_{\epsilon/16}^{\mathcal{A}}].
\end{aligned}$$

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

$$\Pr[\mathcal{B} \text{ guesses correctly} \mid T = e(g_1, g_1)^{\alpha \cdot s} \wedge \neg \mathsf{Box}^{\mathcal{A}}_{\epsilon/16}] \geq \frac{1}{2} - \mathrm{negl}(\lambda). \tag{3}$$

Finally, we also have that

$$\Pr\left[\mathsf{Box}^{\mathcal{A}}_{\kappa} \mid T = e(g_1, g_1)^{\alpha \cdot s}\right] = \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\mathcal{A},\kappa}(\lambda, n+1) = \rho.$$

Combining above fact with Equations 1, 2 and 3, 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 2 is at least $\rho \cdot \kappa - \mathrm{negl}(\lambda)$. This completes the proof. ∎

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

**Observation 4.2.** 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 $\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot i/2n}(\lambda, i) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot(i+1)/2n}(\lambda, i+1) > 0$ and

$$\sum_{i \in \Gamma} \left(\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot i/2n}(\lambda, i) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot(i+1)/2n}(\lambda, i+1)\right) \geq \Pr\text{-}\mathsf{G\text{-}D}_{\mathcal{A},n,\epsilon}(\lambda) - \mathrm{negl}(\lambda).$$

The next lemma will prove that $\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}(i+1)$ and $\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{leq}}(i)$ are approximately equal.

**Lemma 4.5.** 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 $\epsilon(\lambda) > 1/p(\lambda)$,

$$\Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{leq}}_{\epsilon-\frac{\epsilon\cdot(i+1)}{2n}+\frac{\epsilon}{4n}}(\lambda, i) \leq \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\frac{\epsilon\cdot(i+1)}{2n}}(\lambda, i+1) + \mathrm{negl}(\lambda).$$

*Proof.* Let $\rho_1(\lambda) = \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{leq}}_{\epsilon-\frac{\epsilon\cdot(i+1)}{2n}+\frac{\epsilon}{4n}}(\lambda, i)$ and $\rho_2(\lambda) = \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\frac{\epsilon\cdot(i+1)}{2n}}(\lambda, i+1)$.

Let $\mathsf{Expt}_1$ denote the first scenario, and $\mathsf{Expt}_2$ 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 $\mathsf{Hyb}_1$:** This is identical to $\mathsf{Expt}_1$, 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 $\gamma_1$.

**Hybrid $\mathsf{Hyb}_2$** This is identical to $\mathsf{Hyb}_1$, 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 $\gamma_2$.

**Claim 4.5.** Assuming the subgroup decision assumption (Assumption 1), 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 $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_1(\lambda) - \rho_{\mathsf{Hyb}_1}(\lambda) \leq \mathrm{negl}_1(\lambda)$.

*Proof.* Suppose $\rho_1 - \rho_{\mathsf{Hyb}_1} = \eta$ for some non-negligible function $\eta$. 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 $\gamma$-$\mathsf{Dist}^{\mathsf{less}}$ 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 4.1.

■

**Claim 4.6.** Assuming the subgroup decision assumption (Assumption 1), 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 $\eta$. 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 $j \leq i$, 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 $\mathsf{Hyb}_1$ and $\mathsf{Hyb}_2$ 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 4.1.

■

**Claim 4.7.** Assuming the subgroup decision assumption (Assumption 1), for every PPT adversary $\mathcal{A}$, polynomials $n(\cdot), p(\cdot)$ and non-negligible function $\epsilon(\cdot)$, there exists a negligible function $\mathrm{negl}_3(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\rho_{\mathsf{Hyb}_2}(\lambda) - \rho_2(\lambda) \leq \mathrm{negl}_3(\lambda)$.

*Proof.* Suppose $\rho_{\mathsf{Hyb}_2} - \rho_2 = \eta$ for some non-negligible function $\eta$. 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 $j \leq i$. 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 $\gamma$-$\mathsf{Dist}^{\mathsf{less}}$ 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 4.1.

■

■

**Lemma 4.6.** For any PPT adversary $\mathcal{A}$, polynomial $n(\cdot)$ and non-negligible function $\epsilon(\cdot)$, all $\lambda \in \mathbb{N}$,

$$\Pr\text{-}\mathsf{Gap}_{\epsilon/4n}(\lambda, i) \geq \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{less}}_{\epsilon - \epsilon \cdot i/2n}(\lambda, i) - \Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathsf{leq}}_{\epsilon - \epsilon \cdot i/2n - \epsilon/4n}(\lambda, i).$$

*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

$$\Pr\text{-}\mathsf{Gap}_{\mathcal{A},n,\epsilon/4n}(\lambda, i) \geq \max_{\delta \in [0,1/2]} \Pr\left[\begin{array}{l} D \text{ is } \delta\text{-}\mathsf{Dist}^{\mathsf{less}} \wedge \\ D \text{ is not } (\delta - \frac{\epsilon}{4n})\text{-}\mathsf{Dist}^{\mathsf{leq}} \end{array} \ : \ (D, m_0, m_1) \leftarrow \mathsf{MakeBox}_{\mathcal{A},n}(\lambda, i)\right].$$

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

$$\Pr\text{-}\mathsf{Gap}_{\epsilon/4n}(\lambda, i) \geq \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{less}}_{\epsilon-\epsilon\cdot i/2n}(\lambda, i) - \Pr\text{-}\mathsf{Good\text{-}Dec}^{\mathsf{leq}}_{\epsilon-\epsilon\cdot i/2n-\epsilon/4n}(\lambda, i).$$

This concludes the proof. ∎

#### 4.2.5 Proof of Theorem 4.4

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 $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$,

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

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 $\mathsf{Tr}$ 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). 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 $\mathsf{Tr}'$ is defined similar to $\mathsf{Tr}$, except we say that $\mathsf{Tr}'$ occurs if $D$ is $\epsilon$-$\mathsf{Dist}$ box and $\rho_{\mathsf{less}} - \rho_{\mathsf{leq}} > \epsilon/4n$. Note that the only difference between $\mathsf{Tr}$ and $\mathsf{Tr}'$ is that in $\mathsf{Tr}$, 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

$$\begin{aligned}
\Pr[\mathsf{Cor\text{-}Tr}] &= \Pr[\mathsf{Tr}] - \Pr[\mathsf{Tr} \wedge \text{Trace outputs } i \notin S] \\
&= \Pr[\mathsf{Tr}] - \Pr[\mathsf{Fal\text{-}Tr}] \\
&\geq \Pr[\mathsf{Tr}] - \mathrm{negl}_1(\lambda) \text{ (using Theorem 4.2)} \\
&\geq \Pr[\mathsf{Tr}'] - 2^{-O(\lambda)} - \mathrm{negl}_1(\lambda) \\
&\geq \sum_i \frac{\Pr\text{-}\mathsf{Gap}_{\mathcal{A},n,\epsilon/4n}(\lambda, i)}{n} - \mathrm{negl}(\lambda).
\end{aligned}$$

This concludes the proof.

# 5 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 4.1, 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.

## 5.1 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 5.1** (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 $(\epsilon, \delta)$-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 $\epsilon$ and $\delta$ 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 5.1. 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.[5] 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 $q(D) = \frac{\sum_{x \in D} q(x)}{n}$.

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 5.2** (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 $\alpha$-accurate for $\mathcal{Q}$ on $D$ if

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

A randomized algorithm $A : \mathcal{X}^n \to \mathcal{S}$ is said to be an $(\alpha, \beta)$-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 $\alpha$ and $\beta$ 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 $\lambda$; 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 5.3** (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 $\mathsf{Eval}$ runs in time $\mathsf{poly}(\lambda, \log(|X_\lambda|), \log(|Q_\lambda|))$.

## 5.2 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.

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

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).

### 5.2.1 Hardness from Risky Traitor Tracing

**Theorem 5.1.** 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) $(\epsilon, \delta)$-differentially private, (2) $(\alpha, \beta)$-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 $\mathcal{X}_\lambda^n = \big\{D \ : (\mathsf{msk}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)) \leftarrow \mathsf{Setup}(1^\lambda, 1^n), D = (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)\big\}$. 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 $\mathsf{ct}$, and the output of any query $q_{\mathsf{ct}}$ 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 $(\epsilon, \delta)$-differentially private and $(\alpha, \beta)$-accurate for query space $\mathcal{Q}_\lambda$ on $\mathcal{X}_\lambda^n$. From $(\alpha, \beta)$-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

$$\Pr_{A\text{'s coins}}[|q_{\mathsf{ct}}(A(D)) - q_{\mathsf{ct}}(D)| \leq \alpha] \geq 1 - \beta. \tag{4}$$

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 $\mathsf{sk}$, $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = b$. 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

$$\Pr_{A\text{'s coins}}[\lceil \mathsf{Eval}(A(D), q_{\mathsf{ct}}) \rfloor = b] \geq 1 - \beta. \tag{5}$$

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 $\epsilon$,

$$\Pr\text{-}\mathsf{Good}\text{-}\mathsf{Dec}^{\mathcal{T}}_{\mathcal{B},n,\epsilon}(\lambda) \geq 1 - \beta. \tag{6}$$

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

$$\Pr\text{-}\mathsf{Cor}\text{-}\mathsf{Tr}^{\mathcal{T}}_{\mathcal{B},n,\epsilon}(\lambda) \geq f \cdot (1-\beta) - \mathrm{negl}(\lambda), \tag{7}$$

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

$$\Pr_{\substack{(\mathsf{msk},D)\leftarrow\mathsf{Setup}(1^\lambda,1^n)\\ s\leftarrow A(D) \text{ and } \mathsf{Trace}\text{'s coins}}}\left[\mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk})\in[n]\right]\geq \Pr\text{-}\mathsf{Cor}\text{-}\mathsf{Tr}^{\mathcal{T}}_{\mathcal{B},n,\epsilon}(\lambda)\geq f\cdot(1-\beta)-\mathrm{negl}(\lambda). \tag{8}$$

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

$$\Pr_{\substack{(\mathsf{msk},D)\leftarrow\mathsf{Setup}(1^\lambda,1^n)\\ s\leftarrow A(D) \text{ and } \mathsf{Trace}\text{'s coins}}}\left[\mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk})=i^*\right]\geq\frac{f\cdot(1-\beta)}{n}-\mathrm{negl}(\lambda). \tag{9}$$

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 $\mathsf{msk}, D, i^*$,

$$S_{\mathsf{msk},D,i^*}:=\left\{s:\Pr_{\mathsf{Trace}\text{'s coins}}\left[\mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk})=i^*\right]\geq\frac{f}{n^2}\right\}. \tag{10}$$

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

$$\Pr_{\substack{(\mathsf{msk},D)\leftarrow\mathsf{Setup}(1^\lambda,1^n)\\ A\text{'s coins}}}[A(D)\in S_{\mathsf{msk},D,i^*}]\geq\frac{f\cdot(1-\beta)}{2n}. \tag{11}$$

By $(\epsilon,\delta)$-differential privacy of $A$, we have that

$$\Pr_{\substack{(\mathsf{msk},D)\leftarrow\mathsf{Setup}(1^\lambda,1^n)\\ A\text{'s coins}}}[A(D_{-i^*})\in S_{\mathsf{msk},D,i^*}]\geq e^{-\epsilon}\left(\frac{f\cdot(1-\beta)}{2n}-\delta\right)\geq\frac{e^{-\epsilon}\cdot f\cdot(1-\beta)}{4n}. \tag{12}$$

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

$$\Pr_{\substack{(\mathsf{msk},D)\leftarrow\mathsf{Setup}(1^\lambda,1^n)\\ s\leftarrow A(D_{-i^*}) \text{ and } \mathsf{Trace}\text{'s coins}}}\left[\mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk})=i^*\right]\geq\frac{e^{-\epsilon}\cdot f\cdot(1-\beta)}{4n}\times\frac{f}{n^2}. \tag{13}$$

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

$$\Pr\text{-}\mathsf{Fal}\text{-}\mathsf{Tr}^{\mathcal{T}}_{\mathcal{B}^*,n,1/2}(\lambda)\geq\frac{e^{-\epsilon}\cdot f^2\cdot(1-\beta)}{4n^3}. \tag{14}$$

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. ∎

### 5.2.2 Hardness from Risky Traitor Tracing with Singular Trace

**Theorem 5.2.** 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) $(\epsilon, \delta)$-differentially private, (2) $(\alpha, \beta)$-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 5.1, 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 (7). 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

$$\Pr_{\substack{(\mathsf{msk}, D) \leftarrow \mathsf{Setup}(1^\lambda, 1^n) \\ s \leftarrow A(D) \text{ and } \mathsf{Trace}\text{'s coins}}} \left[ \mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk}) = i^* \right] \geq f \cdot (1-\beta) - \mathrm{negl}(\lambda). \tag{15}$$

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 10, 11 and 12 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

$$\Pr_{\substack{(\mathsf{msk}, D) \leftarrow \mathsf{Setup}(1^\lambda, 1^n) \\ s \leftarrow A(D_{-i^*}) \text{ and } \mathsf{Trace}\text{'s coins}}} \left[ \mathsf{Trace}^{\mathsf{Eval}(s,\cdot)}(\mathsf{msk}) = i^* \right] \geq \frac{e^{-\epsilon} \cdot f \cdot (1-\beta)}{4} \times \frac{f}{n}. \tag{16}$$

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

$$\Pr\text{-}\mathsf{Fal}\text{-}\mathsf{Tr}^{\mathcal{T}}_{\mathcal{B}^*, n, 1/2}(\lambda) \geq \frac{e^{-\epsilon} \cdot f^2 \cdot (1-\beta)}{4n}. \tag{17}$$

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

### 5.2.3 Hardness from Subgroup Decision Assumptions

Combining Theorem 5.2 with Theorems 4.2, 4.3 and 4.4, we get the following corollary.

**Corollary 5.1.** Assuming subgroup decision (Assumption 1) and subgroup hiding in target group assumptions (Assumption 2), 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) $(\epsilon, \delta)$-differentially private, (2) $(\alpha, \beta)$-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 $\delta \leq (1-\beta)/4n$.

# 6 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.

## 6.1 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 Test-Good-$A$ and Test-Good-$B$ (defined in Figure 6 and Figure 7) which take as input $r \in \mathcal{M}$ and a ciphertext, and outputs 0/1.

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

**Inputs**: $r \in \mathcal{M}$, ciphertext $\mathsf{ct}_B$
**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 6: Routine Test-Good-$A(r, \mathsf{ct}_B)$

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

**Inputs**: $r \in \mathcal{M}$, ciphertext $\mathsf{ct}_A$
**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 7: Routine Test-Good-$B(r, \mathsf{ct}_A)$

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

**Trace attempt using $\mathsf{Trace}_A$ :**

1. The trace algorithm first searches for an $r_A \in \mathcal{M}$ and ciphertext $\mathsf{ct}_B$ such that Test-Good-$A(r_A, \mathsf{ct}_B) = 1$. 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 Test-Good-$A(r^i, \mathsf{ct}^i) = 1$. 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 $\mathsf{Trace}_A$.

2. Consider the following pirate decoder $D_A$ for $\mathcal{T}_A$. The decoder has $\mathsf{ct}_B$ hardwired. On input ciphertext $\mathsf{ct}_A$, 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 $\mathsf{Trace}_B$.

**Trace attempt using $\mathsf{Trace}_B$** : This is similar to the trace attempt using $\mathsf{Trace}_B$, except that the trace algorithm now builds a pirate decoding box for $\mathsf{Trace}_B$.

1. The trace algorithm searches for an $r_B \in \mathcal{M}$ and ciphertext $\mathsf{ct}_A$ 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 $\mathsf{Trace}_B$.
2. Consider the following pirate decoder $D_B$ for $\mathcal{T}_B$. The decoder has $\mathsf{ct}_A$ hardwired. On input ciphertext $\mathsf{ct}_B$, 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 $\mathcal{T}_A$ and $\mathcal{T}_B$.

## 6.2 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 6.1.** 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 $\eta$ 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 $\mathsf{mpk}_A$ 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 $\mathsf{ct}_A$ 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 $\eta$. ∎

### False-Trace Probability

**Lemma 6.2.** 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 :

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

Clearly, $\Pr\text{-}\mathsf{Fal}\text{-}\mathsf{Tr}_{\mathcal{A},n,\epsilon} = \Pr[\mathcal{F}_1] + \Pr[F_2]$. 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 6.1.** 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 $\epsilon$, $\eta$ such that $\Pr[\mathcal{F}_1] \geq \eta(\lambda)$. We will show that there exists a non-negligible function $\epsilon_A$ 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 $\mathsf{mpk}_A$ 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 $\mathsf{sk}_{\mathcal{A},i}$, 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 $(r_A, \mathsf{ct}_B)$ be the first such pair. The reduction algorithm uses $(r_A, \mathsf{ct}_B)$ 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 $\mathsf{ct}_B$ hardwired, and it takes as input a ciphertext $\mathsf{ct}$ 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 6.2.** 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 $\mathrm{negl}_1(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr[\mathcal{F}_2] \leq \mathrm{negl}_1(\lambda)$.

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 $\mathsf{Trace}_A$ 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 $(\mathsf{mpk} = (\mathsf{mpk}_A, \mathsf{mpk}_B)$, $\mathsf{msk} = (\mathsf{msk}_A, \mathsf{msk}_B)$, $(\mathsf{sk}_1 = (\mathsf{sk}_{A,1}, \mathsf{sk}_{B,1})$, …, $\mathsf{sk}_n = (\mathsf{sk}_{A,n}, \mathsf{sk}_{B,n})))$ $\leftarrow \mathsf{Setup}(1^\lambda, 1^n)$ and sends $\mathsf{mpk}$ to $\mathcal{A}$.
2. Adversary queries for secret keys. For each queried index $i \in \{1, 2, \ldots, n\}$, the challenger sends $\mathsf{sk}_i$.
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 $\mathsf{Trace}_A$. 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 $(r_A, \mathsf{ct}_B)$ be the first such pair.

(b) Challenger uses $(r_A, \mathsf{ct}_B)$ 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 $\mathsf{ct}_B$ hardwired, and it takes as input a ciphertext $\mathsf{ct}$ 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 $\mathsf{Trace}_B$. 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 $(r_B, \mathsf{ct}_A)$ be the first such pair.

   (b) Challenger uses $(r_B, \mathsf{ct}_A)$ 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 $\mathsf{ct}_A$ hardwired, and it takes as input a ciphertext $\mathsf{ct}$ 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 $\lambda$. For simplicity of notations, we will skip the the dependence on $\mathcal{A}$, $n$ and $\epsilon$.

- $\mathsf{Good\text{-}Decoder}$ : $D$ distinguishes between encryptions of $m_0$ and $m_1$ with advantage $\epsilon$
- $\mathsf{Quit}_A$ : No $(r_A, \mathsf{ct}_B)$ 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 $\epsilon/4$
- $\mathsf{Trace}_A\text{-}\mathsf{Succ}$ : $z_A \in S$
- $\mathsf{Trace}_A\text{-}\mathsf{Fail}$ : $\mathsf{Quit}_A$ or $z_A = \perp$
- $\mathsf{Quit}_B$ : No $(r_B, \mathsf{ct}_A)$ 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 6.1.** 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 $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr[\mathcal{E}_1] \geq f_A(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder}] - \mathrm{negl}(\lambda)$.

*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 6.3.** 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 $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\epsilon(\lambda) > 1/p(\lambda)$, $\Pr[\mathcal{E}_1] \geq f_A(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder} \wedge \overline{\mathsf{Quit}_A} \wedge \mathsf{Good\text{-}Decoder}_A] - \mathrm{negl}(\lambda)$.

*Proof.* We will use the PPT adversary $\mathcal{A}$ to build a reduction algorithm $\mathcal{B}$ that interacts with a $\mathcal{T}_A$ 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 $\mathsf{mpk}_A$ 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 $\mathsf{sk}_{\mathcal{A},i}$, 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 $\mathsf{Test\text{-}Good\text{-}}A(r^i, \mathsf{ct}_B^i) = 1$. If no pair exists, it outputs an empty decoding box. Else, let $(r_A, \mathsf{ct}_B)$ be the first such pair. The reduction algorithm uses $(r_A, \mathsf{ct}_B)$ 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 $\mathsf{ct}_B$ hardwired, and it takes as input a ciphertext $\mathsf{ct}$ 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 $\mathrm{negl}(\cdot)$,

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

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

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

■

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 $\lambda$. First, note that

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

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

*Proof.* Let $\mathcal{R}_{\mathsf{Enc}_A}$ and $\mathcal{R}_{\mathsf{Enc}_B}$ denote the space of random coins used by $\mathsf{Enc}_A$ and $\mathsf{Enc}_B$ 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 $\epsilon$, 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 $\mathsf{ct}_B$.

**Claim 6.4.** There exists a negligible function $\mathrm{negl}(\cdot)$ such that for every decoder $D$, all $\lambda \in \mathbb{N}$, $(r, r') \in \mathsf{Good}_D$,

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

The proof of above claim follows from Chernoff bounds and is similar to the proof of Lemma 4.1. Now, we analyse $\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Quit}_A]$. Note that the event $\mathsf{Quit}_A$ 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:

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

$\mathsf{Test\text{-}Good\text{-}}A\mathsf{\text{-}Fail}$ : $\mathsf{Good\text{-}Decoder} \wedge$ Some sample $(r, r') \in \mathsf{Good} \wedge \mathsf{Test\text{-}Good\text{-}}A$ 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 $\lambda$. Next, from Claim 6.4, it follows that there exists a negligible function $\mathrm{negl}_2(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[\mathsf{Test\text{-}Good\text{-}}A\mathsf{\text{-}Fail}] \leq \mathrm{negl}_2(\lambda)$. Thus, we get that $\Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Quit}_A] \leq \mathrm{negl}(\lambda)$. ■

**Lemma 6.4.** There exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

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

*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 $(\epsilon/4)$-$\mathsf{Dist}$ 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 $\mathsf{Trace}_A$, 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 $T \cdot 2^{-O(\lambda)} = \mathrm{negl}(\lambda)$. Thus, we can conclude that $\Pr[\overline{\mathsf{Quit}_A} \wedge \overline{\mathsf{Good\text{-}Decoder}_A}] \leq \mathrm{negl}(\lambda)$. 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 6.2.** 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 $\epsilon(\lambda) \geq 1/p(\lambda)$,

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

*Proof.* The proof of this theorem will follow a similar structure as the proof of Theorem 6.1. 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 6.5.** 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 $\lambda \in \mathbb{N}$, $\Pr[\mathcal{E}_2] \geq f_B(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\mathsf{\text{-}Fail} \wedge \overline{\mathsf{Quit}_B} \wedge \mathsf{Good\text{-}Decoder}_B] - \mathrm{negl}(\lambda)$.

*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 $\mathsf{mpk}_B$ 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 $\mathsf{Trace}_A$. It performs Trace attempt using $\mathsf{Trace}_A$ (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 $(r_B, \mathsf{ct}_A)$ be the first such pair. The reduction algorithm uses $(r_B, \mathsf{ct}_A)$ 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 $\mathsf{ct}_A$ hardwired, and it takes as input a ciphertext $\mathsf{ct}$ 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 6.5.** 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 6.3, 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 $\mathsf{Good}_D$ for a good decoder is at least $\epsilon/2$. For completing this argument, the following observation will be important.

**Observation 6.1.** 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 $\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\text{-}\mathsf{Fail} \wedge \mathsf{Quit}_B$ into two sub-events as before and argue that they both occur with at most negligible probability. ∎

**Lemma 6.6.** There exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

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

The proof of this lemma is identical to that of Lemma 6.4. Combining the above lemmas, we get that there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[\mathcal{E}_2] \geq f_B(\lambda, n) \cdot \Pr[\mathsf{Good\text{-}Decoder} \wedge \mathsf{Trace}_A\text{-}\mathsf{Fail}] - \mathrm{negl}(\lambda)$.

■

Combining Theorem 6.1 and Theorem 6.2, we can compute the 'riskyness' of our traitor tracing scheme. Concretely, we get that

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

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.

[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.

[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 CryptologyCRYPTO'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.

[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.