


Ad Hoc (Decentralized) Broadcast, Trace, and Revoke

Ji Luo 

Paul G. Allen School of Computer Science & Engineering,
University of Washington, Seattle, USA
`luoji@cs.washington.edu`

Abstract. Traitor tracing schemes [Chor–Fiat–Naor, Crypto ’94] help content distributors fight against piracy and are defined with the content distributor as a trusted authority having access to the secret keys of all users. While the traditional model caters well to its original motivation, its centralized nature makes it unsuitable for many scenarios. For usage among mutually untrusted parties, a notion of *ad hoc* traitor tracing (naturally with the capability of broadcast and revocation) is proposed and studied in this work. Such a scheme allows users in the system to generate their own public/secret key pairs, without trusting any other entity. To encrypt, a list of public keys is used to identify the set of recipients, and decryption is possible with a secret key for any of the public keys in the list. In addition, there is a tracing algorithm that given a list of recipients’ public keys and a pirate decoder capable of decrypting ciphertexts encrypted to them, identifies at least one recipient whose secret key must have been used to construct the said decoder.

Two constructions are presented. The first is based on functional encryption for circuits (conceptually, obfuscation) and has constant-size ciphertext, yet its decryption time is linear in the number of recipients. The second is a generic transformation that reduces decryption time at the cost of increased ciphertext size. A lower bound on the trade-off between ciphertext size and decryption time is shown, indicating that the two constructions achieve all possible optimal trade-offs, i.e., they fully demonstrate the Pareto front of efficiency. The lower bound also applies to broadcast encryption (hence all mildly expressive attribute-based encryption schemes) and is of independent interest.

1 Introduction

Traitor tracing schemes [22] enable content distributors to fight against piracy. A content distributor such as a media streaming service can generate a public key and many different secret keys for individual subscribers, all of which can decrypt the ciphertexts created using the public key. Given a pirate decoder capable of decrypting, which could have been created from the secret keys of

To reviewers having seen this paper on ePrint: The submitted version is **newer**. This research is “open-thoughts”. See <https://github.com/GeeLaw/ahbtr>.

multiple subscribers, the tracing algorithm can find at least one subscriber (a traitor) whose key was used to create the said decoder. A long line of subsequent works [10,11,13,16,20,35–38,55,64,67] proposed the different security definitions, extended the functionality, and presented new constructions.

While the traditional model caters well to the needs of content distributors, its centralized nature makes it unsuitable for many scenarios, e.g., when a group of individuals want to communicate among themselves and trace traitors who provide decoders to outsiders. For example, in an encrypted group chat among protesters [67], the users are worried about potential infiltration by government agents. To mitigate this concern, they want to trace traitors and remove them from the group. If they used a traditional traitor tracing scheme, whoever set it up would be able to impersonate anyone since they would know all the secret keys. Moreover, as words are spread and the protest gets wider support, more people need to join the group. The joining process should as simple as possible, preferably without interaction. This motivation naturally calls for a decentralized notion of traitor tracing, termed *ad hoc* traitor tracing in this work.

The first question is thus naturally the following:

What is the right notion of a secure ad hoc traitor tracing scheme?

Having formalized its syntax and security, we study its constructions:

*How can such a scheme be constructed,
from what assumptions, and with what efficiency?*

Efficiency improvement never ends until we reach the optimum, for which it is necessary to understand where the limit stands:

What bounds are there on the efficiency of such schemes?

Our Contributions. We provide answers to all three questions:

- *Conceptually*, we pose the question of *ad hoc* traitor tracing, develop from the ideas thereof, and eventually arrive at the definitions for *ad hoc* broadcast, trace, and revoke (AH-BTR). We prove the relation among the security notions considered in this work.
- *Construction-wise*, we present secure AH-BTR schemes based on functional encryption for general circuits [12]. With polynomial factors in the security parameter ignored, they achieve

$$\begin{array}{ll} \text{encryption time} & T_{\text{Enc}} = O(N), \\ \text{ciphertext size} & |\text{ct}| = O(N^{1-\gamma}), \\ \text{decryption time} & T_{\text{Dec}} = O(N^\gamma), \end{array}$$

for any constant $0 \leq \gamma \leq 1$.

- *Questing for the ultimate efficiency*, we prove that for all secure AH-BTR,

$$|\text{ct}| \cdot T_{\text{Dec}} = \Omega(N),$$

so our schemes offer all possible optimal trade-offs between $|\text{ct}|$ and T_{Dec} , fully demonstrating the Pareto front of AH-BTR efficiency. Better yet, the bound holds for a restricted kind of weakly secure broadcast encryption [28], which is a specific case of attribute-based encryption [39,58]. Our result is the first time-space lower bound applicable to any computationally secure BE scheme,¹ shedding new insights into the efficiency of ABE and BE.

A final addition is that our scheme is *compatible* with the existing public-key encryption schemes, i.e., the keys of such a scheme can be those of any secure public-key encryption, and there is no need to regenerate keys to take advantage of our scheme.

Open Questions. The tracing model in this work is black-box and classical, and recent works [66,67] have studied white-box or quantum traitor tracing. Conceptually, it is interesting to understand the *ad hoc* versions of those tracing models.

Another question for future investigation is whether (weakened versions of) AH-BTR can be constructed from more lightweight assumptions, e.g., factoring-related, group-based, or lattice-based assumptions. Potential relaxations include making the scheme bounded,² settling for static security, considering security against bounded collusion, and only achieving threshold tracing [53].

Organization. In Sect. 1.1, we provide an overview of our results. In Sect. 2, we present the preliminaries.³ In Sect. 3, we formally define *ad hoc* broadcast, trace, and revoke (AH-BTR) and its security notions, and prove the relation among them. In Sect. 4, we define *ad hoc* private linear broadcast encryption (AH-PLBE), an intermediate object for constructing AH-BTR, and construct such a scheme. In Sect. 5, we present the construction of AH-BTR from AH-PLBE. In Sect. 6, we show how to trade ciphertext size for decryption time in AH-BTR. In Sect. 7, we prove the lower bound of the trade-offs between ciphertext size and decryption time.

1.1 Overview

Developing Definitions. We start with the first principles of *ad hoc* traitor tracing. Syntactically, there should be a key generation algorithm that is run by each user of the system.⁴ To encrypt, a list of public keys is used to identify the

¹ Previous works [4,7,24,30,43,47,51] show a few efficiency lower bounds related to ABE and BE. Yet they only apply to information-theoretically secure primitives and even specific construction techniques. Moreover, all of them are space (ciphertext or secret key size, or their trade-off) lower bounds. Indeed, based on obfuscation [15] or both LWE and pairing [2], BE with $|\text{ct}|, |\text{sk}| = O(1)$ can be constructed.

² A maximum of number of recipients per ciphertext is set when generating a key pair, and only “compatible” public keys can be used to form a recipient set.

³ The technical parts of the preliminaries (those beyond the opening paragraphs) are only needed for the constructions. Sections 3 and 7 do not depend on them.

⁴ We aim for a scheme without any trusted party, so there should be no global set-up.

set of recipients. Decryption should only require one secret key from the list of public keys. In addition, the decryptor gets random access to all the recipients' public keys as well as the ciphertext. The choice to give random access to these inputs is based on performance concerns, as the decryptor might not have to read all of the public keys or the ciphertext.

It should be clear that such a scheme would automatically have the functionality of broadcast encryption [28].⁵ There is no event prior to encryption that “binds” the system to a specific, fixed set of possible recipients, and the encryptor is free to use whatever public keys it sees fit. Similarly, the encryptor can remove any public key when it encrypts a second ciphertext, i.e., the scheme supports revocation. Therefore, the object is named *ad hoc* broadcast, trace, and revoke (AH-BTR).

As usual with broadcast encryption, we do not hide the list of recipients. Hiding the recipients makes ciphertext grow at least linearly with the number of recipients, diminishing the potential of efficiency. As we shall see, it is possible to construct AH-BTR with short ciphertexts.

Due to the decentralized nature of such systems, an adversary might indistinguishably generate malformed keys, which could potentially evade tracers that only take well-formed keys into account. To make it worse, a malformed key could be used to mount a denial-of-service attack against (other) honest users if it appears in the list of recipients' public keys during encryption — the encryption algorithm might have been carelessly designed and the presence of certain malformed keys could make it impossible to decrypt for anyone, including the recipients with honestly generated public keys.

In order to protect against such attacks by definition, we require correctness be *robust* against malformed keys — however, for performance reasons, namely to be able to index into any particular public key in constant time, we reject *blatantly* malformed keys, e.g., those of incorrect lengths, in the definition of correctness. This restriction does not hamper the usefulness of such a scheme.

As for security, when attacking the *traceability* of the scheme, the adversary is allowed to supply an arbitrary list of recipients' public keys, generated honestly by the challenger or (adversarially) by the adversary, so that the definition covers the scenario when (blatantly or not) malformed keys are present in the list of recipients' public keys. The tracing algorithm is given oracle access to a stateless⁶ decoder. It must *not accuse* an honest user, defined as one whose public key is generated by the challenger without its secret key revealed to the adversary. It *must find* a traitor as long as the decoder has sufficient advantage (i.e., succeeds in decrypting with sufficient probability), where *traitors* are associated with public keys in the recipient list that are either generated by the challenger with

⁵ Decentralized versions of broadcast encryption were studied in [27,56] with interactive management of recipient sets. *Ad hoc* (threshold) broadcast encryption was studied in [25,61] with constructions for bounded schemes requiring global set-up.

⁶ The general transformation [11,45] to deal with stateful decoder applies to our definition of AH-BTR, *mutatis mutandis*.

their secret keys revealed to the adversary or crafted by the adversary in any manner (e.g., skewed distribution, or even without a well-defined secret key).

Once the issues above are identified and conceptually resolved (as done here), it is straight-forward to define AH-BTR analogously to traditional broadcast, revoke, and trace schemes [38,52,54].

Simplifying Security Notions. Traditionally [11], traceability has been defined using one comprehensive interactive experiment,⁷ which is complicated to work with. Intuitively, the notion requires that *i*) a traitor should be found from a decoder with sufficient advantage and *ii*) no honest user should be identified as a traitor, regardless of the decoder advantage.

We thus define two security notions for AH-BTR capturing the requirements separately. The former is called *completeness* and the latter is called *soundness*. Their conjunction is equivalent to *traceability*. Since only one requirement is considered in each notion, both of them can be vastly simplified and the interactions in those notions are minimal. They are also more convenient for reductionist proofs.

Construction. Our first construction of AH-BTR follows the existing blueprint of traitor tracing schemes from private linear broadcast encryption (PLBE) schemes introduced in [11]. We first define an *ad hoc* version of PLBE:⁸

- Everyone generates their own public and secret key pair $(\mathbf{pk}, \mathbf{sk})$.
- Encryption uses a list $\{\mathbf{pk}_j\}_{j \in [N]}$ of N public keys of the recipients as well as a cut-off index $0 \leq i_\perp \leq N$.
- Decryption is possible with \mathbf{sk}_j if $j > i_\perp$.

There are two security requirements. Message-hiding requires that the plaintext is hidden if $i_\perp = N$. Index-hiding requires that an adversary without \mathbf{sk}_j for an *honest* \mathbf{pk}_j cannot distinguish between cut-off index being $(j - 1)$ versus j .

Colloquially, the cut-off index i_\perp disables $\mathbf{sk}_1, \dots, \mathbf{sk}_{i_\perp}$, and the only way to detect whether an index is disabled is to have control over the corresponding key pair (by knowing \mathbf{sk} or generating a malformed \mathbf{pk}). When $i_\perp = N$, the plaintext should be hidden since all keys are disabled.

Given an AH-PLBE scheme, an AH-BTR scheme can be constructed by adapting the work of [11]. The AH-BTR inherits key generation and decryption algorithms from AH-PLBE. To perform AH-BTR encryption, simply encrypt using AH-PLBE with $i_\perp = 0$, disabling no key so that every recipient can decrypt. Given a pirate decoder with advantage at least ε , the tracing algorithm computes its advantages with the cut-off index i_\perp being $0, 1, 2, \dots, N$, and identifies the recipient associated with \mathbf{pk}_{i^*} as a traitor if the advantage changes by $\Omega(\varepsilon/N)$ when i_\perp increases from $(i^* - 1)$ to i^* .

⁷ While some previous works [8,36,64] separate traceability into multiple notions, those notions still share one single complicated security experiment.

⁸ AH-PLBE can be cast as multi-authority attribute-based encryption [19] for 1-local monotone functions without global set-up.

The message-hiding property translates to completeness, and index-hiding to soundness. It now remains to construct an AH-PLBE.

Constructing AH-PLBE. It is folklore that any public-key encryption (PKE) scheme can be used to construct a naïve PLBE by encrypting individually to each recipient. The individual ciphertext that corresponds to a disabled key encrypts garbage instead of the actual plaintext. This scheme is also *ad hoc*. The downside of it is that the ciphertext is of size $\Omega(N)$.

Our scheme uses obfuscation to compress the naïve PLBE ciphertext. The ciphertext will contain an obfuscated program, which, when evaluated at $j \in [N]$, allows us to recover the PKE ciphertext under pk_j . However, the obfuscated program itself cannot simply compute each PKE ciphertext if we want AH-PLBE ciphertexts of size $\text{o}(N)$, as there is not enough space in the program to encode all the public keys that have been independently generated.

Laconic oblivious transfer (OT) [21] comes to rescue. It allows compressing an arbitrarily long string D down to a fixed-length hash h with which one can efficiently perform oblivious transfer. The sender can encrypt messages L_0, L_1 to a hash h and an index m into D . The time to encrypt is independent of the length of D . The receiver will be able to obtain $L_{D[m]}$ by decrypting the laconic OT ciphertext.

During AH-PLBE encryption, we use laconic OT to compress the list of public keys. The obfuscated program in our AH-PLBE ciphertext, when evaluated at $j \in [N]$, will output *i*) a garbled circuit whose input (resp. output) is a PKE public key (resp. ciphertext) and *ii*) a bunch of laconic OT ciphertexts that decrypts to the labels so that the garbled circuit is evaluated at pk_j . Decryption proceeds in the obvious manner.

The obfuscated program size, thus the ciphertext size, can be made constant,⁹ because both the time to garble a PKE encryption circuit and the time of laconic OT encryptions are constant.

You Can (Not) Optimize. While our basic construction enjoys constant-size ciphertext, its decryption algorithm runs in time $\Omega(N)$. Concretely, the laconic OT hash is a Merkle tree, and before performing laconic OT decryption, it is necessary to reconstruct the tree as it is not stored in the ciphertext. In contrast, the decryption time of the scheme implied by the naïve PLBE is constant in the RAM model, as it only looks at the relevant piece of the underlying PKE ciphertext.

We can trade ciphertext size for decryption time by using the naïve PLBE on top of our construction. By grouping the recipients into $\Theta(N^{1-\gamma})$ sets of size $\Theta(N^\gamma)$ and using our basic construction over each set, we obtain a scheme with ciphertext size $\Theta(N^{1-\gamma})$ and decryption time $\Theta(N^\gamma)$. The core idea of this transformation was formalized as the user expansion compiler [64] in the context of traditional traitor tracing.

⁹ We ignore *fixed* polynomial factors in the security parameter. The point is that the size does not grow with N (the number of recipients).

All the constructions we now know have $|\text{ct}| \cdot T_{\text{Dec}} = \Omega(N)$, where $|\text{ct}|$ is the ciphertext length and T_{Dec} is the decryption time. It turns out that this bound necessarily holds for all secure AH-BTR, and the blame is on the functionality of broadcast encryption (not traitor tracing). Indeed, it is possible to make both $|\text{ct}|$ and T_{Dec} constant in a traditional traitor tracing scheme [16]. In existing broadcast encryption (or revocation) schemes [1,2,9,16,18,26,31] for N users, encrypting to arbitrary subsets of size S or $(N - S)$ makes $|\text{ct}| \cdot T_{\text{Dec}} = \Omega(S)$. It is precisely the capability to encrypt to many $\Theta(N)$ -subsets among N users that is the deal breaker, as we shall see in the formal proof. Interestingly, the adversary used in the proof is simply the decryption algorithm, so the bound holds as long as the scheme is not *blatantly* insecure.

We explain the ideas of our proof based on a corollary¹⁰ of a result [60] dealing with random oracles in the presence of non-uniform advice. Let $S, T \geq 0$ be such that $ST \ll N$. The corollary says that for any adversary learning any S -bit function (advice) of a random string $R \xleftarrow{\$} \{0, 1\}^N$ and additionally (adaptively) querying at most T bits in R , it is “indistinguishable” to flip a bit in R at a random location after the advice is computed (using the non-flipped R) and before queries are answered, even if the index of the potentially flipped bit is revealed to the adversary after the advice is computed.

Back to AH-BTR. Imagine that there are $2N$ users in the system, associated with key pairs $(\text{pk}_{j,s}, \text{sk}_{j,s})$ for $j \in [N]$ and $s \in \{0, 1\}$. Consider a ciphertext ct encrypting a random plaintext to $\{\text{pk}_{j,R[j]}\}_{j \in [N]}$ for a random string R and regard ct as the advice. Let’s try decrypting ct using $\text{sk}_{i^*, R[i^*]}$ for a random $i^* \xleftarrow{\$} [N]$. Each time the AH-BTR decryption algorithm queries pk_j , we probe $R[j]$ and respond with $\text{pk}_{j,R[j]}$. By way of contradiction, suppose $|\text{ct}| \cdot T_{\text{Dec}} \ll N$, which would translate to the setting of the corollary as $S = |\text{ct}|$, $T \leq T_{\text{Dec}}$, and $ST \ll N$.

By the correctness of AH-BTR, the attempted decryption should successfully recover the plaintext. From the corollary it follows that flipping $R[i^*]$ should also lead to successful recovery. But if $R[i^*]$ is flipped after ct is computed, by the security of AH-BTR, the attempted decryption must fail to recover the plaintext except for negligible probability, yielding a contradiction.

2 Preliminaries

We denote by $\lambda \in \mathbb{N}$ the security parameter, by $\text{poly}(\cdot)$ a polynomial function, and by $\text{negl}(\lambda)$ a negligible function of λ . Efficient algorithms are probabilistic random-access machines $M^w(x)$ of running time $\text{poly}(|x|, |w|)$. Efficient adversaries (in interactive experiments) are probabilistic Turing machines of (total) running time $\text{poly}(\lambda)$, with or without $\text{poly}(\lambda)$ -long advices. (All of the proofs in this work are uniform.) The advantage of \mathcal{A} in distinguishing Exp_0 and Exp_1

¹⁰ This corollary is also a lower bound of a probabilistic variant of Yao’s box problem [63] (generalized and studied in [23]), on which our proof can be alternatively based.

is $\Pr[\text{Exp}_0^A(1^\lambda) = 1] - \Pr[\text{Exp}_1^A(1^\lambda) = 1]$. We write $\approx, \approx_s, \equiv$ for computational indistinguishability, statistical indistinguishability, and identity.

Under the standard assumption that a pseudorandom generator (with polynomial security) exists, we can assume, whenever convenient, that a randomized algorithm uses a uniformly random λ -bit string as its randomness (without losing polynomial security considered in this work or degrading its efficiency).

For $n, n' \in \mathbb{N}$, we write $[n..n']$ for the set $\{n, \dots, n'\}$, and $[n]$ for $[1..n]$. For a bit-string D , we denote by $|D|$ its bit-length, and given an index $m \in [|D|]$, we denote by $D[m]$ the m^{th} bit of D . For two bit-strings D, D' , their concatenation is $D \| D'$. Given a circuit $C : \{0, 1\}^{n+M_0} \rightarrow \{0, 1\}^{n'}$ and $w \in \{0, 1\}^n$, we define $C[w]$ to be C with w hardwired as its first portion of input, so $C[w](x) = C(w \| x)$. For an event X , its indicator random variable is $\mathbb{1}_X$. For events X, Y in the same probability space, “ X implies Y ” means $X \subseteq Y$.

Garbled Circuits. The following version of partially hiding garbling [40] suffices for the purpose of this work.

Definition 1 (garbled circuit [6,40,49,62]). A circuit garbling scheme consists of 2 efficient algorithms:

- $\text{Garble}(1^\lambda, C, w)$ takes as input a circuit $C : \{0, 1\}^{n+M_0} \rightarrow \{0, 1\}^{n'}$ and some hardwired input $w \in \{0, 1\}^n$. It outputs a garbled circuit \hat{C} and M_0 pairs of labels $L_{m_0,b} \in \{0, 1\}^\lambda$ for $m_0 \in [M_0]$, $b \in \{0, 1\}$.
- $\text{Eval}(1^\lambda, \hat{C}, x, \{L_{m_0}\}_{m_0 \in [M_0]})$ takes as input a garbled circuit, a non-hardwired input, and M_0 labels. It outputs an n' -bit string.

The scheme must be correct, i.e., for all $\lambda \in \mathbb{N}$, $n, M_0, n' \in \mathbb{N}$, $w \in \{0, 1\}^n$, $C : \{0, 1\}^{n+M_0} \rightarrow \{0, 1\}^{n'}$, $x \in \{0, 1\}^{M_0}$,

$$\Pr \left[\begin{array}{l} (\hat{C}, \{L_{m_0,b}\}_{m_0 \in [M_0], b \in \{0,1\}}) \xleftarrow{s} \text{Garble}(1^\lambda, C, w) \\ : \text{Eval}(1^\lambda, \hat{C}, x, \{L_{m_0,x[m_0]}\}_{m_0 \in [M_0]}) = C[w](x) \end{array} \right] = 1.$$

Definition 2 (garbled circuit security [6,40,49,62]). Let $(\text{Garble}, \text{Eval})$ be a circuit garbling scheme (Definition 1). A simulator is an efficient algorithm

$$\begin{aligned} & \text{SimGarble}(1^\lambda, C : \{0, 1\}^{n+M_0} \rightarrow \{0, 1\}^{n'}, x \in \{0, 1\}^{M_0}, y \in \{0, 1\}^{n'}) \\ & \rightarrow (\hat{C}, \{L_{m_0}\}_{m_0 \in [M_0]}) \end{aligned}$$

taking as input a circuit, a non-hardwired input, and a circuit output, and producing a simulated garbled circuit and M_0 simulated labels. The scheme is w -hiding (or secure for the purpose of this work) if there exists a simulator SimGarble such that $\text{Exp}_{\text{GC}}^0 \approx \text{Exp}_{\text{GC}}^1$, where $\text{Exp}_{\text{GC}}^b(1^\lambda)$ with adversary \mathcal{A} proceeds as follows:

- **Challenge.** Launch $\mathcal{A}(1^\lambda)$ and receive a circuit $C : \{0, 1\}^{n+M_0} \rightarrow \{0, 1\}^{n'}$, a hardwired input $w \in \{0, 1\}^n$, and a non-hardwired input $x \in \{0, 1\}^{M_0}$ from

it. Run

if $b = 0$, $(\hat{C}, \{L_{m_0,b}\}_{m_0 \in [M_0], b \in \{0,1\}}) \xleftarrow{\$} \text{Garble}(1^\lambda, C, w);$
if $b = 1$, $(\hat{C}, \{L_{m_0,x[m_0]}\}_{m_0 \in [M_0]}) \xleftarrow{\$} \text{SimGarble}(1^\lambda, C, x, C[w](x));$

and send $(\hat{C}, \{L_{m_0,x[m_0]}\}_{m_0 \in [M_0]})$ to \mathcal{A} .

– **Guess.** \mathcal{A} outputs a bit b' , which is the output of the experiment.

Puncturable Pseudorandom Function. We rely on PPRF [14,17,44,57].

Definition 3 (PPRF [14,17,44,57]). A puncturable pseudorandom function (PPRF) family (with key space, domain, and codomain $\{0,1\}^\lambda$) consists of 2 efficient algorithms:

- $\text{Puncture}(1^\lambda, k \in \{0,1\}^\lambda, x)$ takes as input a non-punctured key and a point. It outputs a punctured key \mathring{k}_x .
- $\text{Eval}(1^\lambda, k, x \in \{0,1\}^\lambda)$ takes as input a (punctured or non-punctured) key and a point. It is deterministic and outputs a λ -bit string.

The scheme must be correct, i.e., for all $\lambda \in \mathbb{N}$, $x, x' \in \{0,1\}^\lambda$ such that $x \neq x'$,

$$\Pr \left[k \xleftarrow{\$} \{0,1\}^\lambda, \mathring{k}_x \xleftarrow{\$} \text{Puncture}(1^\lambda, k, x) : \text{Eval}(1^\lambda, k, x') = \text{Eval}(1^\lambda, \mathring{k}_x, x') \right] = 1.$$

Definition 4 (PPRF security [14,17,44,57]). A PPRF ($\text{Puncture}, \text{Eval}$) per Definition 3 is pseudorandom at the punctured point (or secure for the purpose of this work) if $\text{Exp}_{\text{PPRF}}^0 \approx \text{Exp}_{\text{PPRF}}^1$, where $\text{Exp}_{\text{PPRF}}^b(1^\lambda)$ with adversary \mathcal{A} proceeds as follows:

– **Challenge.** Launch $\mathcal{A}(1^\lambda)$ and receive from it a point $x \in \{0,1\}^\lambda$. Run

$k \xleftarrow{\$} \{0,1\}^\lambda, \mathring{k}_x \xleftarrow{\$} \text{Puncture}(1^\lambda, k, x), r_0 \xleftarrow{\$} \text{Eval}(1^\lambda, k, x), r_1 \xleftarrow{\$} \{0,1\}^\lambda,$

and send (\mathring{k}_x, r_b) to \mathcal{A} .

– **Guess.** \mathcal{A} outputs a bit b' , which is the output of the experiment.

Public-Key Encryption. Our *ad hoc* broadcast, trace, and revoke scheme can be based on any public-key encryption scheme.

Definition 5 (PKE). A public-key encryption (PKE) scheme (with message space $\{0,1\}^\lambda$ and public key length $M_0(\lambda)$) consists of 3 efficient algorithms:

- $\text{Gen}(1^\lambda)$ outputs a pair (pk, sk) of public and secret keys with $|\text{pk}| = M_0(\lambda)$.
- $\text{Enc}(1^\lambda, \text{pk}, \mu \in \{0,1\}^\lambda)$ takes as input the public key and a message. It outputs a ciphertext ct .
- $\text{Dec}(1^\lambda, \text{sk}, \text{ct})$ takes as input the secret key and a ciphertext. It outputs a message.

The scheme must be correct, i.e., for all $\lambda \in \mathbb{N}$, $\mu \in \{0, 1\}^\lambda$,

$$\Pr \left[\begin{array}{c} (\text{pk}, \text{sk}) \xleftarrow{\$} \text{Gen}(1^\lambda) \\ \text{ct} \xleftarrow{\$} \text{Enc}(1^\lambda, \text{pk}, \mu) \end{array} : \text{Dec}(1^\lambda, \text{sk}, \text{ct}) = \mu \right] = 1.$$

Definition 6 (PKE security). A PKE scheme $(\text{Gen}, \text{Enc}, \text{Dec})$ per Definition 5 is semantically secure for random messages (or secure for the purpose of this work) if

$$\{(1^\lambda, \mu_0, \mu_1, \text{pk}, \text{ct}_0)\} \approx \{(1^\lambda, \mu_0, \mu_1, \text{pk}, \text{ct}_1)\},$$

where $(\text{pk}, \text{sk}) \xleftarrow{\$} \text{Gen}(1^\lambda)$ and $\mu_b \xleftarrow{\$} \{0, 1\}^\lambda$, $\text{ct}_b \xleftarrow{\$} \text{Enc}(1^\lambda, \text{pk}, \mu_b)$ for $b \in \{0, 1\}$.

Laconic Oblivious Transfer. We rely on laconic oblivious transfer [21].

Definition 7 (laconic OT [21]). A laconic oblivious transfer (OT) scheme (with message space $\{0, 1\}^\lambda$) consists of 4 efficient algorithms:

- $\text{Gen}(1^\lambda, M \in \mathbb{N})$ takes the database length as input and outputs a hash key hk .
- $\text{Hash}(1^\lambda, \text{hk}, D \in \{0, 1\}^M)$ takes as input a hash key and a database. It is deterministic, runs in time $O(M) \text{poly}(\lambda, \log M)$, and outputs a hash h of length $\text{poly}(\lambda, \log M)$ and a processed database \hat{D} .
- $\text{Send}(1^\lambda, \text{hk}, h, m \in [M], L_0 \in \{0, 1\}^\lambda, L_1 \in \{0, 1\}^\lambda)$ takes as input a hash key, a hash, an index, and two labels (messages). It outputs a ciphertext ct .
- $\text{Recv}^{\hat{D}}(1^\lambda, \text{hk}, h, m \in [M], \text{ct})$ is given random access to a processed database, and takes as input a hash key, a hash, an index, and a ciphertext. It runs in time $\text{poly}(\lambda, \log M)$ and outputs a label (message).

The scheme must be correct, i.e., for all $\lambda \in \mathbb{N}$, $M \in \mathbb{N}$, $D \in \{0, 1\}^M$, $m \in [M]$, $L_0, L_1 \in \{0, 1\}^\lambda$,

$$\Pr \left[\begin{array}{c} \text{hk} \xleftarrow{\$} \text{Gen}(1^\lambda, M) \\ (h, \hat{D}) \leftarrow \text{Hash}(1^\lambda, \text{hk}, D) \\ \text{ct} \xleftarrow{\$} \text{Send}(1^\lambda, \text{hk}, h, m, L_0, L_1) \end{array} : \text{Recv}^{\hat{D}}(1^\lambda, \text{hk}, h, m, \text{ct}) = L_{D[m]} \right] = 1.$$

We only need database-selective security [3]. The following indistinguishability-based definition is equivalent to the usual simulation-based formulation.

Definition 8 (laconic OT security [3, 21, 46]). A laconic OT scheme $(\text{Gen}, \text{Hash}, \text{Send}, \text{Recv})$ per Definition 7 is database-selectively sender-private (or secure for the purpose of this work) if $\text{Exp}_{\text{LOT}}^0 \approx \text{Exp}_{\text{LOT}}^1$, where $\text{Exp}_{\text{LOT}}^b(1^\lambda)$ with adversary \mathcal{A} proceeds as follows:

- **Setup.** Launch $\mathcal{A}(1^\lambda)$ and receive from it some $M \in \mathbb{N}$ and a database $D \in \{0, 1\}^M$. Run

$$\text{hk} \xleftarrow{\$} \text{Gen}(1^\lambda, M), \quad (h, \hat{D}) \leftarrow \text{Hash}(1^\lambda, \text{hk}, D),$$

and send hk to \mathcal{A} .

- **Challenge.** \mathcal{A} submits an index $m \in [M]$ into D and two labels (messages) $L_0, L_1 \in \{0, 1\}^\lambda$. Run

$$\text{ct} \xleftarrow{\$} \begin{cases} \text{Send}(1^\lambda, \text{hk}, h, m, L_0, L_1), & \text{if } b = 0; \\ \text{Send}(1^\lambda, \text{hk}, h, m, L_{D[m]}, L_{D[m]}), & \text{if } b = 1; \end{cases}$$

and send ct to \mathcal{A} .

- **Guess.** \mathcal{A} outputs a bit b' , which is the output of the experiment.

Obfuscation. We rely on indistinguishability obfuscator for polynomial-sized domain.

Definition 9 ((circuit) obfuscator [5]). A (circuit) obfuscator is an efficient algorithm $\text{Obf}(1^\lambda, C)$ taking a circuit $C : \{0, 1\}^n \rightarrow \{0, 1\}^{n'}$ as input and producing a circuit $\tilde{C} : \{0, 1\}^n \rightarrow \{0, 1\}^{n'}$ as output. The scheme must be correct, i.e., for all $\lambda \in \mathbb{N}$, $n, n' \in \mathbb{N}$, $C : \{0, 1\}^n \rightarrow \{0, 1\}^{n'}$, $x \in \{0, 1\}^n$,

$$\Pr[\text{Obf}(1^\lambda, C)(x) = C(x)] = 1.$$

Definition 10 ($i\mathcal{O}$ [5] for $\text{poly}(\lambda)$ -sized domain). An obfuscator Obf (Definition 9) is an indistinguishability obfuscator for polynomial-sized domain ($i\mathcal{O}$ for $\text{poly}(\lambda)$ -sized domain) if $\text{Exp}_{i\mathcal{O}}^0 \approx \text{Exp}_{i\mathcal{O}}^1$, where $\text{Exp}_{i\mathcal{O}}^b(1^\lambda)$ with adversary \mathcal{A} proceeds as follows:

- **Challenge.** Launch $\mathcal{A}(1^\lambda)$ and receive from it the domain size 1^{2^n} and two circuits $C_0, C_1 : \{0, 1\}^n \rightarrow \{0, 1\}^{n'}$. Send $\text{Obf}(1^\lambda, C_b)$ to \mathcal{A} .
- **Guess.** \mathcal{A} outputs a bit b' . The output of the experiment is b' if C_0, C_1 have the same (description) size and $C_0(x) = C_1(x)$ for all $x \in \{0, 1\}^n$. Otherwise, the output is set to 0.

Assumption. All of the primitives defined in this section are implied by the existence of weakly selectively secure, single key, and sublinearly succinct public-key functional encryption for general circuits (so-called *obfuscation-minimum PKFE*), of which we refer the reader to [46] for the precise definition.

Lemma 1. Suppose there exists an obfuscation-minimum PKFE with polynomial security, then there exist

- [6,49,62] a secure circuit garbling scheme (Definitions 1 and 2),
- [14,17,33,44] a secure PPRF (Definitions 3 and 4),
- [folklore] a secure PKE scheme (Definitions 5 and 6),
- [3,21,46,50] a secure laconic OT scheme (Definitions 7 and 8), and
- [48,50] an $i\mathcal{O}$ for $\text{poly}(\lambda)$ -sized domain (Definitions 9 and 10),

with polynomial security.

Alternatively, those primitives can be based on the existence of $i\mathcal{O}$ and one-way function. However, $i\mathcal{O}$ security (for circuits whose domains are not necessarily $\text{poly}(\lambda)$ -sized) is not known to be *falsifiable* [32] and it is hard to conceive [29] a reduction of $i\mathcal{O}$ security to *complexity assumptions* [34]. Since all of the security notions defined in this section are falsifiable, it is unsatisfactory to base them on $i\mathcal{O}$ from a theoretical point of view.

In contrast, obfuscation-minimum PKFE security is falsifiable and there are constructions [41,42] from well-studied complexity assumptions. The point of Lemma 1 is to base our constructions solely on one falsifiable assumption, or even complexity assumptions.

3 *Ad Hoc* Broadcast, Trace, and Revoke

This section concerns the definitions for *ad hoc* broadcast, trace, and revoke. After formally defining the syntax and correctness of AH-BTR, we present an intuitive definition of its security. While the security definition is comprehensive, it is not the easiest to work with, so we turn to define two simpler security notions, whose conjunction is equivalent to the comprehensive definition. The proof of their equivalence follows the definitions. Later in this paper, we will only work with the simpler notions.

Definition 11 (AH-BTR). *An ad hoc broadcast, trace, and revoke (AH-BTR) scheme (with message space $\{0,1\}^\lambda$ and public key length $M_0(\lambda)$) consists of 4 efficient algorithms:*

- $\text{Gen}(1^\lambda)$ outputs a pair (pk, sk) of public and secret keys with $|\text{pk}| = M_0(\lambda)$.
- $\text{Enc}(1^\lambda, \{\text{pk}_j\}_{j \in [N]}, \mu \in \{0,1\}^\lambda)$ takes as input a list of public keys and a message. It outputs a ciphertext ct .
- $\text{Dec}^{\{\text{pk}_j\}_{j \in [N]}, \text{ct}}(1^\lambda, N, i \in [N], \text{sk}_i)$ is given random access to a list of public keys and a ciphertext, and takes as input the length of the list, an index, and a secret key. It outputs a message.
- $\text{Trace}^{\mathcal{D}}(1^\lambda, \{\text{pk}_j^*\}_{j \in [N]}, 1^{1/\epsilon^*})$ is given oracle access to a (stateless randomized) distinguisher \mathcal{D} and takes as input a list of public keys and an error bound. It outputs an index $i^* \in \{\perp\} \cup [N]$.

The scheme must be robustly correct, i.e., for all $\lambda \in \mathbb{N}$, $N \in \mathbb{N}$, $i \in [N]$, $\{\text{pk}_j\}_{j \in [N] \setminus \{i\}}$ ¹¹ such that $|\text{pk}_j| = M_0(\lambda)$ for all $j \in [N] \setminus \{i\}$, and $\mu \in \{0,1\}^\lambda$,

$$\Pr \left[\begin{array}{l} (\text{pk}_i, \text{sk}_i) \xleftarrow{\$} \text{Gen}(1^\lambda) \\ \text{ct} \xleftarrow{\$} \text{Enc}(1^\lambda, \{\text{pk}_j\}_{j \in [N]}, \mu) \end{array} : \text{Dec}^{\{\text{pk}_j\}_{j \in [N]}, \text{ct}}(1^\lambda, N, i, \text{sk}_i) = \mu \right] = 1.$$

Definition 12 (traceability). *An AH-BTR scheme $(\text{Gen}, \text{Enc}, \text{Dec}, \text{Trace})$ per Definition 11 is traceable if all efficient adversary wins $\text{Exp}_{\text{trace}}$ only with negligible probability, where $\text{Exp}_{\text{trace}}(1^\lambda)$ with adversary \mathcal{B} proceeds as follows:*

¹¹ These public keys could be out of the support of Gen , i.e., malformed.

- **Setup.** Launch $\mathcal{B}(1^\lambda)$. Initialize the set S to \emptyset and let $Q \leftarrow 0$.
- **Query.** Repeat the following for arbitrarily many rounds determined by \mathcal{B} . In each round, \mathcal{B} has two options:
 - \mathcal{B} can request that a new user be initialized and obtain the newly created public key. Upon this request, let $Q \leftarrow Q + 1$, run

$$(\text{pk}_Q, \text{sk}_Q) \xleftarrow{\$} \text{Gen}(1^\lambda),$$
 insert Q into S , and send pk_Q to \mathcal{B} .
 - \mathcal{B} can query for sk_t by submitting $t \in [Q]$. Upon this query, remove t from S and send sk_t to \mathcal{B} .
- **Challenge.** \mathcal{B} outputs a (probabilistic) circuit \mathcal{D} , a list $\{\text{pk}_j^*\}_{j \in [N]}$ of public keys, and an error bound $1^{1/\varepsilon^*}$. Run

$$i^* \xleftarrow{\$} \text{Trace}^{\mathcal{D}}(1^\lambda, \{\text{pk}_j^*\}_{j \in [N]}, 1^{1/\varepsilon^*}).$$

Let

- **FalsePos** be the event that $i^* \in [N]$ and $\text{pk}_{i^*}^* = \text{pk}_s$ for some $s \in S$,
- **GoodDist** the event that

$$\left| \Pr \left[\begin{array}{l} \mu_0 \xleftarrow{\$} \{0, 1\}^\lambda, \quad \mu_1 \xleftarrow{\$} \{0, 1\}^\lambda \\ \beta \xleftarrow{\$} \{0, 1\} \\ \text{ct} \xleftarrow{\$} \text{Enc}(1^\lambda, \{\text{pk}_j^*\}_{j \in [N]}, \mu_\beta) \end{array} : \mathcal{D}(\mu_0, \mu_1, \text{ct}) = \beta \right] - \frac{1}{2} \right| \geq \varepsilon^*,$$

- and **NotFound** the event that $i^* \notin [N]$ (i.e., $i^* = \perp$).

\mathcal{B} wins if and only if $\text{FalsePos} \vee (\text{GoodDist} \wedge \text{NotFound})$.

AH-BTR as defined above is KEM, following [64]. Using hybrid encryption, such a scheme can be easily adapted for arbitrarily long messages with traceability under adversarially chosen messages. As noted in Remark 3 of [65], traceability implies KEM security (or IND-CPA when combined with hybrid encryption).

3.1 Simplified Security Notions

The traceability of AH-BTR guarantees that a traitor must be found (if the decoder is good enough) and innocent users must not be accused (whether or not the decoder is good enough). Decomposing the two requirements (plus some apparent weakening) makes each of them simpler (in particular, non-interactive). The first requirement is called *completeness*, and the second *soundness*.

Definition 13 (completeness). An AH-BTR scheme $(\text{Gen}, \text{Enc}, \text{Dec}, \text{Trace})$ per Definition 11 is complete if all efficient adversary wins $\text{Exp}_{\text{complete}}$ only with negligible probability, where $\text{Exp}_{\text{complete}}(1^\lambda)$ with adversary \mathcal{C} proceeds as follows:

- **Challenge.** Launch $\mathcal{C}(1^\lambda)$, which outputs a (probabilistic) circuit \mathcal{D} , a list $\{\text{pk}_j^*\}_{j \in [N]}$ of public keys, and an error bound $1^{1/\varepsilon^*}$. Run

$$i^* \xleftarrow{\$} \text{Trace}^{\mathcal{D}}(1^\lambda, \{\text{pk}_j^*\}_{j \in [N]}, 1^{1/\varepsilon^*}).$$

Let

- **GoodDist** be the event that

$$\Pr \left[\begin{array}{l} \mu_0 \xleftarrow{\$} \{0,1\}^\lambda, \quad \mu_1 \xleftarrow{\$} \{0,1\}^\lambda \\ \beta \xleftarrow{\$} \{0,1\} \\ \text{ct} \xleftarrow{\$} \text{Enc}(1^\lambda, \{\text{pk}_j^*\}_{j \in [N]}, \mu_\beta) \end{array} : \mathcal{D}(\mu_0, \mu_1, \text{ct}) = \beta \right] - \frac{1}{2} \geq \varepsilon^*,$$

- and **NotFound** the event that $i^* \notin [N]$ (i.e., $i^* = \perp$).

\mathcal{C} wins if and only if $\text{GoodDist} \wedge \text{NotFound}$.

Definition 14 (soundness). An AH-BTR scheme $(\text{Gen}, \text{Enc}, \text{Dec}, \text{Trace})$ per Definition 11 is sound if all efficient adversary wins $\text{Exp}_{\text{sound}}$ only with negligible probability, where $\text{Exp}_{\text{sound}}(1^\lambda)$ with adversary \mathcal{C} proceeds as follows:

- **Challenge.** Run $(\text{pk}, \text{sk}) \xleftarrow{\$} \text{Gen}(1^\lambda)$, then run $\mathcal{C}(1^\lambda, \text{pk})$, which outputs a (probabilistic) circuit \mathcal{D} , some $N \in \mathbb{N}$, a challenge index $i_\perp^* \in [N]$, a list $\{\text{pk}_j^*\}_{j \in [N] \setminus \{i_\perp^*\}}$ of public keys, and an error bound $1^{1/\varepsilon^*}$. Let $\text{pk}_{i_\perp^*}^* \leftarrow \text{pk}$ and run

$$i^* \xleftarrow{\$} \text{Trace}^{\mathcal{D}}(1^\lambda, \{\text{pk}_j^*\}_{j \in [N]}, 1^{1/\varepsilon^*}).$$

\mathcal{C} wins if and only if $i^* = i_\perp^*$ (the event **FalsePos**).

Theorem 2 (¶). An AH-BTR scheme is traceable if and only if it is both complete and sound.

Proof (Theorem 2). The reductionist proof of necessity is straight-forward — the query phase is unused by the reduction algorithm for completeness, and used only for creating the public key given to the adversary as input for soundness.

To show sufficiency, suppose the AH-BTR scheme $(\text{Gen}, \text{Enc}, \text{Dec}, \text{Trace})$ is both complete and sound and let \mathcal{B} be an efficient adversary against its traceability. We consider two efficient adversaries. \mathcal{C}_1 is against the completeness of the scheme. It works by internally simulating the traceability game for \mathcal{B} and outputting whatever \mathcal{B} outputs. Consider the coupling between $\text{Exp}_{\text{complete}}$ for \mathcal{C}_1 and the simulated $\text{Exp}_{\text{trace}}$ for \mathcal{B} inside, writing the events for adversary \mathcal{X} in its security experiment with subscript \mathcal{X} ,

$$\text{GoodDist}_{\mathcal{C}_1} \iff \text{GoodDist}_{\mathcal{B}} \quad \text{and} \quad \text{NotFound}_{\mathcal{C}_1} \iff \text{NotFound}_{\mathcal{B}}.$$

Therefore,

$$\Pr[\text{GoodDist}_{\mathcal{B}} \wedge \text{NotFound}_{\mathcal{B}}] = \Pr[\text{GoodDist}_{\mathcal{C}_1} \wedge \text{NotFound}_{\mathcal{C}_1}].$$

\mathcal{C}_2 is against the soundness of the scheme. Let $B = \text{poly}(\lambda) > 1$ be an upper bound of the running time of \mathcal{B} . The adversary \mathcal{C}_2 does the following:

- $\mathcal{C}_2(\text{pk})$ launches \mathcal{B} , initializes S to \emptyset , lets $Q \leftarrow 0$, and samples and sets

$$s^* \xleftarrow{\$} [B], \quad \text{pk}_{s^*} \leftarrow \text{pk}, \quad (\text{pk}_t, \text{sk}_t) \xleftarrow{\$} \text{Gen}() \quad \text{for } t \in [B] \setminus \{s^*\}.$$

- \mathcal{C}_2 answers queries from \mathcal{B} and updates Q, S as stipulated by the query phase of the traceability experiment, except that it aborts if \mathcal{B} queries for \mathbf{sk}_{s^*} .
- After the query phase, \mathcal{B} outputs

$$\mathcal{D}, \quad \{\mathbf{pk}_j^*\}_{j \in [N]}, \quad 1^{1/\varepsilon^*},$$

and \mathcal{C}_2 samples or sets

$$i_\perp^* \begin{cases} \xleftarrow{s} I_\perp^*, & \text{if } I_\perp^* \leftarrow \{i \in [N] : \mathbf{pk}_i^* = \mathbf{pk}\} \neq \emptyset; \\ \leftarrow \perp & \text{otherwise.} \end{cases}$$

It aborts if $i_\perp^* = \perp$. Otherwise, \mathcal{C}_2 outputs

$$\mathcal{D}, \quad N, \quad i_\perp^*, \quad \{\mathbf{pk}_j^*\}_{j \in [N] \setminus \{i_\perp^*\}}, \quad 1^{1/\varepsilon^*}.$$

Consider the coupling between $\text{Exp}_{\text{sound}}$ for \mathcal{C}_2 and the simulated $\text{Exp}_{\text{trace}}$ for \mathcal{B} inside. Routine calculation yields

$$\Pr[\text{FalsePos}_{\mathcal{C}_2}] \geq \frac{1}{B^2} \Pr[\text{FalsePos}_{\mathcal{B}}].$$

By the union bound,

$$\begin{aligned} & \Pr[\text{FalsePos}_{\mathcal{B}} \vee (\text{GoodDist}_{\mathcal{B}} \wedge \text{NotFound}_{\mathcal{B}})] \\ & \leq \Pr[\text{FalsePos}_{\mathcal{B}}] + \Pr[\text{GoodDist}_{\mathcal{B}} \wedge \text{NotFound}_{\mathcal{B}}] \\ & \leq B^2 \Pr[\text{FalsePos}_{\mathcal{C}_2}] + \Pr[\text{GoodDist}_{\mathcal{C}_1} \wedge \text{NotFound}_{\mathcal{C}_1}] \\ & = (\text{poly}(\lambda))^2 \text{negl}(\lambda) + \text{negl}(\lambda) = \text{negl}(\lambda). \end{aligned} \quad \square$$

4 *Ad Hoc* Private Linear Broadcast Encryption

Our construction of AH-BTR follows that of traitor tracing schemes in [11]. We define *ad hoc* private broadcast linear encryption (AH-PLBE) by adapting the notion of PLBE [11] to the *ad hoc* setting.

Definition 15 (AH-PLBE). An *ad hoc* private linear broadcast encryption (AH-PLBE) scheme (with message space $\{0, 1\}^\lambda$ and public key length $M_0(\lambda)$) consists of 3 efficient algorithms:

- $\text{Gen}(1^\lambda)$ outputs a pair $(\mathbf{pk}, \mathbf{sk})$ of public and secret keys with $|\mathbf{pk}| = M_0(\lambda)$.
- $\text{Enc}(1^\lambda, \{\mathbf{pk}_j\}_{j \in [N]}, i_\perp \in [0..N], \mu \in \{0, 1\}^\lambda)$ takes as input a list of public keys, a cut-off index, and a message. It outputs a ciphertext ct .
- $\text{Dec}^{\{\mathbf{pk}_j\}_{j \in [N]}, \text{ct}}(1^\lambda, N, i \in [N], \mathbf{sk}_i)$ is given random access to a list of public keys and a ciphertext, and takes as input the length of the list, an index, and a secret key. It outputs a message.

The scheme must be robustly correct, i.e., for all $\lambda \in \mathbb{N}$, $N \in \mathbb{N}$, $i \in [N]$, $\{\mathbf{pk}_j\}_{j \in [N] \setminus \{i\}}$ ¹² such that $|\mathbf{pk}_j| = M_0(\lambda)$ for all $j \in [N] \setminus \{i\}$, and $\mu \in \{0, 1\}^\lambda$,

$$\Pr \left[\begin{array}{c} (\mathbf{pk}_i, \mathbf{sk}_i) \xleftarrow{s} \text{Gen}(1^\lambda) \\ \text{ct} \xleftarrow{s} \text{Enc}(1^\lambda, \{\mathbf{pk}_j\}_{j \in [N]}, 0, \mu) \end{array} : \text{Dec}^{\{\mathbf{pk}_j\}_{j \in [N]}, \text{ct}}(1^\lambda, N, i, \mathbf{sk}_i) = \mu \right] = 1.$$

¹² These public keys could be out of the support of Gen , i.e., malformed.

Security. We define security notions of AH-PLBE analogously to those in [11], except “mode indistinguishability” (Game 1 in [11]), which is not needed here. The two security definitions have a one-to-one correspondence to the simplified security notions of AH-BTR in Sect. 3.1. Namely, *message-hiding* translates to *completeness*, and *index-hiding* translates to *soundness*.

Definition 16 (message-hiding). An AH-PLBE scheme $(\text{Gen}, \text{Enc}, \text{Dec})$ per Definition 15 is message-hiding if $\text{Exp}_{\text{MH}}^0 \approx \text{Exp}_{\text{MH}}^1$, where $\text{Exp}_{\text{MH}}^b(1^\lambda)$ with adversary \mathcal{A} proceeds as follows:

- **Challenge.** Launch $\mathcal{A}(1^\lambda)$ and receive from it a list $\{\text{pk}_j^*\}_{j \in [N]}$ of public keys. Run

$$\mu_0 \xleftarrow{\$} \{0, 1\}^\lambda, \quad \mu_1 \xleftarrow{\$} \{0, 1\}^\lambda, \quad \text{ct} \xleftarrow{\$} \text{Enc}(1^\lambda, \{\text{pk}_j^*\}_{j \in [N]}, N, \mu_b),$$

and send $(\mu_0, \mu_1, \text{ct})$ to \mathcal{A} .

- **Guess.** \mathcal{A} outputs a bit b' , which is the output of the experiment.

Definition 17 (index-hiding). An AH-PLBE scheme $(\text{Gen}, \text{Enc}, \text{Dec})$ per Definition 15 is index-hiding if $\text{Exp}_{\text{IH}}^0 \approx \text{Exp}_{\text{IH}}^1$, where $\text{Exp}_{\text{IH}}^b(1^\lambda)$ with adversary \mathcal{A} proceeds as follows:

- **Challenge.** Run $(\text{pk}, \text{sk}) \xleftarrow{\$} \text{Gen}(1^\lambda)$, launch $\mathcal{A}(1^\lambda, \text{pk})$, and receive from it some $N \in \mathbb{N}$, a cut-off index $i_\perp^* \in [N]$, and a list $\{\text{pk}_j^*\}_{j \in [N] \setminus \{i_\perp^*\}}$ of public keys. Let $\text{pk}_{i_\perp^*}^* \leftarrow \text{pk}$, run

$$\mu \xleftarrow{\$} \{0, 1\}^\lambda, \quad \text{ct} \xleftarrow{\$} \text{Enc}(1^\lambda, \{\text{pk}_j^*\}_{j \in [N]}, i_\perp^* - 1 + b, \mu),$$

and send (μ, ct) to \mathcal{A} .

- **Guess.** \mathcal{A} outputs a bit b' , which is the output of the experiment.

4.1 Construction

Ingredients of Construction 1. Let

- $\text{GC} = (\text{GC.Garble}, \text{GC.Eval}, \text{GC.SimGarble})$ be a circuit garbling scheme such that GC.Garble uses λ -bit randomness,
- $\text{PPRF} = (\text{PPRF.Puncture}, \text{PPRF.Eval})$ a PPRF,
- $\text{PKE} = (\text{PKE.Gen}, \text{PKE.Enc}, \text{PKE.Dec})$ a PKE scheme such that PKE.Enc uses λ -bit randomness and whose public keys are (exactly) of polynomial length M_0 ,
- $\text{LOT} = (\text{LOT.Gen}, \text{LOT.Hash}, \text{LOT.Send}, \text{LOT.Recv})$ a laconic OT scheme,
- Obf an obfuscator.

Construction 1 (AH-PLBE). Our AH-PLBE works as follows:

- Gen is the same as PKE.Gen .

$C_{GC}[N, \text{hk}, h, i_{\perp}, \mu_{\perp}, \mu, k^{\text{GC}}, k^{\text{PKE}}, \{k_{m_0}^{\text{LOT}}\}_{m_0 \in [M_0]}](i)$	
Hardwired.	N , number of users; hk , laconic OT hash key; h , laconic OT hash of $D = \text{pk}_1 \parallel \dots \parallel \text{pk}_N$; i_{\perp} , cut-off index; μ_{\perp} , cut-off message; μ , message; k^{GC} , PPRF key for circuit garbling; k^{PKE} , PPRF key for public-key encryption; $k_{m_0}^{\text{LOT}}$, PPRF key for sending the m_0^{th} label using laconic OT.
Input.	$i \in [N]$, index of recipient.
Output.	Computed as follows.
	$r_i^{\text{GC}} \leftarrow \text{PPRF.Eval}(k^{\text{GC}}, i)$ $r_i^{\text{PKE}} \leftarrow \text{PPRF.Eval}(k^{\text{PKE}}, i)$ $r_{i, m_0}^{\text{LOT}} \leftarrow \text{PPRF.Eval}(k_{m_0}^{\text{LOT}}, i) \text{ for } m_0 \in [M_0]$ $(\widehat{C}_{\text{ct}, i}, \{L_{i, m_0, b}\}_{m_0 \in [M_0], b \in \{0, 1\}})$ $\leftarrow \begin{cases} \text{GC.Garble}(\widehat{C}_{\text{ct}}, (\mu_{\perp}, r_i^{\text{PKE}}); r_i^{\text{GC}}), & \text{if } i \leq i_{\perp}; \\ \text{GC.Garble}(\widehat{C}_{\text{ct}}, (\mu, r_i^{\text{PKE}}); r_i^{\text{GC}}), & \text{if } i > i_{\perp}; \end{cases}$ $\text{LOT.ct}_{i, m_0} \leftarrow \text{LOT.Send}(\text{hk}, h, (i-1)M_0 + m_0,$ $\quad L_{i, m_0, 0}, L_{i, m_0, 1}; r_{i, m_0}^{\text{LOT}}) \text{ for } m_0 \in [M_0]$ output $(\widehat{C}_{\text{ct}, i}, \{\text{LOT.ct}_{i, m_0}\}_{m_0 \in [M_0]})$
$C_{\text{ct}}[\mu'_i, r_i^{\text{PKE}}](\text{pk}_i)$	
Hardwired.	μ'_i , message or cut-off message; r_i^{PKE} , public-key encryption randomness.
Input.	pk_i , public key of recipient.
Output.	$\text{PKE.ct}_i \leftarrow \text{PKE.Enc}(\text{pk}_i, \mu'_i; r_i^{\text{PKE}}).$

Fig. 1. The circuits C_{GC} and C_{ct} in Construction 1.

- $\text{Enc}(\{\text{pk}_j\}_{j \in [N]}, i_\perp, \mu)$ first checks whether $|\text{pk}_j| = M_0$ for all $j \in [N]$. If not, it outputs $\text{ct} = \perp$ and terminates. Otherwise, the algorithm hashes down the public keys by running

$$\begin{aligned} M &\leftarrow NM_0, & D &\leftarrow \text{pk}_1 \parallel \dots \parallel \text{pk}_N, \\ \text{hk} &\xleftarrow{\$} \text{LOT.Gen}(M), & (h, \widehat{D}) &\leftarrow \text{LOT.Hash}(\text{hk}, D). \end{aligned}$$

It samples the cut-off message $\mu_\perp \xleftarrow{\$} \{0, 1\}^\lambda$ and PPRF keys

$$k^{\text{GC}} \xleftarrow{\$} \{0, 1\}^\lambda, \quad k^{\text{PKE}} \xleftarrow{\$} \{0, 1\}^\lambda, \quad k_{m_0}^{\text{LOT}} \xleftarrow{\$} \{0, 1\}^\lambda \quad \text{for } m_0 \in [M_0],$$

and obfuscates C_{GC} (Fig. 1) by running

$$\widetilde{C}_{\text{GC}} \xleftarrow{\$} \text{Obf}(C_{\text{GC}}[N, \text{hk}, h, i_\perp, \mu_\perp, \mu, k^{\text{GC}}, k^{\text{PKE}}, \{k_{m_0}^{\text{LOT}}\}_{m_0 \in [M_0]}]).$$

The algorithm outputs $\text{ct} = (\text{hk}, \widetilde{C}_{\text{GC}})$ as the ciphertext.

- $\text{Dec}^{\{\text{pk}_j\}_{j \in [N]}, \text{ct}}(N, i, \text{sk}_i)$ first parses $\text{ct} = (\text{hk}, \widetilde{C}_{\text{GC}})$ and recomputes

$$M \leftarrow NM_0, \quad D \leftarrow \text{pk}_1 \parallel \dots \parallel \text{pk}_N, \quad (h, \widehat{D}) \leftarrow \text{LOT.Hash}(\text{hk}, D).$$

The algorithm next runs the obfuscated circuit,

$$(\widehat{C}_{\text{ct}, i}, \{\text{LOT.ct}_{i, m_0}\}_{m_0 \in [M_0]}) \leftarrow \widetilde{C}_{\text{GC}}(i),$$

to obtain the garbled C_{ct} (Fig. 1) for the decryptor and the laconic OT ciphertexts sending its labels. It then receives the labels,

$$L_{i, m_0, \text{pk}_i[m_0]} \leftarrow \text{LOT.Recv}^{\widehat{D}}(\text{hk}, h, (i-1)M_0 + m_0, \text{LOT.ct}_{i, m_0}) \quad \text{for } m_0 \in [M_0],$$

and evaluates the garbled circuit,

$$\text{PKE.ct}_i \leftarrow \text{GC.Eval}(\widehat{C}_{\text{ct}, i}, \text{pk}_i, \{L_{i, m_0, \text{pk}_i[m_0]}\}_{m_0 \in [M_0]}),$$

to obtain the PKE ciphertext under the decryptor's public key. Lastly, the algorithm runs and outputs (as the decrypted message)

$$\mu \leftarrow \text{PKE.Dec}(\text{sk}_i, \text{PKE.ct}_i).$$

Robustness Correctness. This can be verified by inspection.

Efficiency. By the efficiency of laconic OT, LOT.Gen takes time $\text{poly}(\lambda, \log N)$, LOT.Hash takes time $O(N) \text{poly}(\lambda, \log N)$, and $|\text{hk}|, |h| = \text{poly}(\lambda, \log N)$. As we shall see later, it suffices to pad C_{GC} to size $\text{poly}(\lambda, \log N)$ for the security proofs to go through. Putting these together,

$$T_{\text{Enc}} = O(N) \text{poly}(\lambda, \log N), \quad |\text{ct}| = \text{poly}(\lambda, \log N), \quad T_{\text{Dec}} = O(N) \text{poly}(\lambda, \log N).$$

In practice and for security reasons, we always assume $N \leq 2^\lambda$ and $\log N$ is absorbed by λ . Therefore, with $\text{poly}(\lambda)$ factors ignored, both encryption and decryption take linear time, and the ciphertext is constant-size.

Compatibility. Since the key generation algorithm of Construction 1 is just the key generation algorithm of the underlying PKE scheme (which only has to be semantically secure for random messages), it is compatible with the existing public-key encryption schemes, i.e., existing users possessing PKE key pairs can utilize our AH-PLBE without regenerating their keys.

4.2 Message-Hiding Property

Theorem 3 (¶). *Suppose in Construction 1, the obfuscator Obf is an $i\mathcal{O}$ for $\text{poly}(\lambda)$ -sized domain, then the resultant AH-PLBE is message-hiding.*

Proof (Theorem 3). For Construction 1, the only difference between Exp_{MH}^0 and Exp_{MH}^1 is whether C_{GC} used to create $\text{ct} = (\text{hk}, \tilde{C}_{\text{GC}})$ has μ_0 or μ_1 hardwired as μ . In C_{GC} (Fig. 1), μ is used only in the branch $i > i_\perp$, which is never taken in Exp_{MH}^0 or Exp_{MH}^1 because i_\perp is hardwired to be N and the domain of i is $[N]$. Therefore, the two C_{GC} 's in Exp_{MH}^0 and Exp_{MH}^1 being obfuscated are functionally equivalent and have the same size. Moreover, their domain size is N (polynomially large). Therefore, $\text{Exp}_{\text{MH}}^0 \approx \text{Exp}_{\text{MH}}^1$ reduces to the $i\mathcal{O}$ security for $\text{poly}(\lambda)$ -sized domain of Obf. \square

4.3 Index-Hiding Property

Theorem 4 (¶). *Suppose in Construction 1, all of the ingredients are secure, then the resultant AH-PLBE is index-hiding.*

Proof (Theorem 4). The only difference between Exp_{IH}^0 and Exp_{IH}^1 is whether the C_{GC} being obfuscated hardwires μ (in Exp_{IH}^0) or μ_\perp (in Exp_{IH}^1) into C_{ct, i_\perp^*} , which only affects the output of C_{GC} at $i = i_\perp^*$. We consider the following hybrids, each (except the first) described by the changes from the previous one:

- H_0^b (for $b \in \{0, 1\}$) is Exp_{IH}^b , where

$$\begin{aligned} \text{hk} &\stackrel{\$}{\leftarrow} \text{LOT.Gen}(NM_0), & (h, \widehat{D}) &\stackrel{\$}{\leftarrow} \text{LOT.Hash}(\text{hk}, \text{pk}_1^* \parallel \dots \parallel \text{pk}_N^*), \\ k^{\text{GC}} &\stackrel{\$}{\leftarrow} \{0, 1\}^\lambda, & k^{\text{PKE}} &\stackrel{\$}{\leftarrow} \{0, 1\}^\lambda, & k_{m_0}^{\text{LOT}} &\stackrel{\$}{\leftarrow} \{0, 1\}^\lambda \quad \text{for } m_0 \in [M_0], \\ \tilde{C}_{\text{GC}} &\stackrel{\$}{\leftarrow} \text{Obf}(C_{\text{GC}}[N, \text{hk}, h, i_\perp^* - 1 + b, \mu_\perp, \mu, k^{\text{GC}}, k^{\text{PKE}}, \{k_{m_0}^{\text{LOT}}\}_{m_0 \in [M_0]}]), \\ \text{ct} &= (\text{hk}, \tilde{C}_{\text{GC}}). \end{aligned}$$

- H_1^b alters the obfuscation into

$$\begin{aligned} \tilde{C}_{\text{GC}} &\stackrel{\$}{\leftarrow} \text{Obf}(C'_{\text{GC}}[N, \text{hk}, h, \mu_\perp, \mu, \\ &\quad i_\perp^*, k_{i_\perp^*}^{\text{GC}}, k_{i_\perp^*}^{\text{PKE}}, \{k_{m_0, i_\perp^*}^{\text{LOT}}\}_{m_0 \in [M_0]}, \widehat{C}_{\text{ct}, i_\perp^*}, \{\text{LOT.ct}_{i_\perp^*, m_0}\}_{m_0 \in [M_0]}]), \end{aligned}$$

where

- C'_{GC} is defined in Fig. 2,

$C'_{GC}[N, \text{hk}, h, \mu_{\perp}, \mu, i_{\perp}^*, \overset{\circ}{k}_{i_{\perp}^*}^{\text{GC}}, \overset{\circ}{k}_{i_{\perp}^*}^{\text{PKE}}, \{\overset{\circ}{k}_{m_0, i_{\perp}^*}^{\text{LOT}}\}_{m_0 \in [M_0]}, \widehat{C}_{\text{ct}, i_{\perp}^*}, \{\text{LOT.ct}_{i_{\perp}^*, m_0}\}_{m_0 \in [M_0]}](i)$

Hardwired. $N, \text{hk}, h, \mu_{\perp}, \mu,$ see Fig. 1;
 $i_{\perp}^*,$ challenge cut-off index;
 $\overset{\circ}{k}_{\dots, i_{\perp}^*},$ PPRF keys punctured at i_{\perp}^* ;
 $\widehat{C}_{\text{ct}, i_{\perp}^*}, \text{LOT.ct}_{i_{\perp}^*, \dots},$ hardwired output of C'_{GC} at $i = i_{\perp}^*$.

Input. $i \in [N],$ index of recipient.

Output. Computed as follows.

if $i = i_{\perp}^*$:
 output $(\widehat{C}_{\text{ct}, i_{\perp}^*}, \{\text{LOT.ct}_{i_{\perp}^*, m_0}\}_{m_0 \in [M_0]})$ as hardwired
 else:
 $r_i^{\text{GC}} \leftarrow \text{PPRF.Eval}(\overset{\circ}{k}_{i_{\perp}^*}^{\text{GC}}, i)$
 $r_i^{\text{PKE}} \leftarrow \text{PPRF.Eval}(\overset{\circ}{k}_{i_{\perp}^*}^{\text{PKE}}, i)$
 $r_{i, m_0}^{\text{LOT}} \leftarrow \text{PPRF.Eval}(\overset{\circ}{k}_{m_0, i_{\perp}^*}^{\text{LOT}}, i)$ for $m_0 \in [M_0]$
 $(\widehat{C}_{\text{ct}, i}, \{L_{i, m_0, b}\}_{m_0 \in [M_0], b \in \{0, 1\}})$
 $\leftarrow \begin{cases} \text{GC.Garble}(\widehat{C}_{\text{ct}}, (\mu_{\perp}, r_i^{\text{PKE}}); r_i^{\text{GC}}), & \text{if } i < i_{\perp}^*; \\ \text{GC.Garble}(\widehat{C}_{\text{ct}}, (\mu, r_i^{\text{PKE}}); r_i^{\text{GC}}), & \text{if } i > i_{\perp}^*; \end{cases}$
 $\text{LOT.ct}_{i, m_0} \leftarrow \text{LOT.Send}(\text{hk}, h, (i-1)M_0 + m_0,$
 $\qquad\qquad\qquad L_{i, m_0, 0}, L_{i, m_0, 1}; r_{i, m_0}^{\text{LOT}})$ for $m_0 \in [M_0]$
 output $(\widehat{C}_{\text{ct}, i}, \{\text{LOT.ct}_{i, m_0}\}_{m_0 \in [M_0]})$

Fig. 2. The circuit C'_{GC} in the proof of Theorem 4.

- the PPRF keys are punctured at i_{\perp}^* by running

$$\begin{aligned}
 \overset{\circ}{k}_{i_{\perp}^*}^{\text{GC}} &\xleftarrow{\$} \text{PPRF.Puncture}(k^{\text{GC}}, i_{\perp}^*), \\
 \overset{\circ}{k}_{i_{\perp}^*}^{\text{PKE}} &\xleftarrow{\$} \text{PPRF.Puncture}(k^{\text{PKE}}, i_{\perp}^*), \\
 \overset{\circ}{k}_{m_0, i_{\perp}^*}^{\text{LOT}} &\xleftarrow{\$} \text{PPRF.Puncture}(k_{m_0}^{\text{LOT}}, i_{\perp}^*) \quad \text{for } m_0 \in [M_0],
 \end{aligned}$$

- and the hardwired output $(\widehat{C}_{\text{ct}, i_{\perp}^*}, \{\text{LOT.ct}_{i_{\perp}^*, m_0}\}_{m_0 \in [M_0]})$ of C'_{GC} at $i = i_{\perp}^*$ is computed as

$$\begin{aligned}
 r_i^{\text{GC}} &\leftarrow \text{PPRF.Eval}(k^{\text{GC}}, i_{\perp}^*), & r_i^{\text{PKE}} &\leftarrow \text{PPRF.Eval}(k^{\text{PKE}}, i_{\perp}^*), \\
 r_{i_{\perp}^*, m_0}^{\text{LOT}} &\leftarrow \text{PPRF.Eval}(k_{m_0}^{\text{LOT}}, i_{\perp}^*) & \text{for } m_0 &\in [M_0],
 \end{aligned}$$

- $$\begin{aligned}
& (\widehat{C}_{\text{ct}, i_{\perp}^*}, \{L_{i_{\perp}, m_0, b}\}_{m_0 \in [M_0], b \in \{0,1\}}) \\
& \leftarrow \begin{cases} \text{GC.Garble}(C_{\text{ct}}, (\mu_{\perp}, r_{i_{\perp}^*}^{\text{PKE}}); r_{i_{\perp}^*}^{\text{GC}}), & \text{if } b = 0; \\ \text{GC.Garble}(C_{\text{ct}}, (\mu_{\perp}, r_{i_{\perp}^*}^{\text{PKE}}); r_{i_{\perp}^*}^{\text{GC}}), & \text{if } b = 1; \end{cases} \\
& \text{LOT.ct}_{i_{\perp}^*, m_0} \leftarrow \text{LOT.Send}(\text{hk}, h, (i_{\perp}^* - 1)M_0 + m_0, \\
& \quad L_{i_{\perp}, m_0, 0}, L_{i_{\perp}, m_0, 1}; r_{i_{\perp}^*, m_0}^{\text{LOT}}) \quad \text{for } m_0 \in [M_0]. \\
& - \text{H}_2^b \text{ changes } r_{i_{\perp}^*}^{\text{GC}}, r_{i_{\perp}^*}^{\text{PKE}}, \text{ and } r_{i_{\perp}^*, m_0}^{\text{LOT}} \text{'s into true randomness, i.e.,} \\
& \quad r^{\text{GC}} \xleftarrow{\$} \{0, 1\}^{\lambda}, \quad r^{\text{PKE}} \xleftarrow{\$} \{0, 1\}^{\lambda}, \quad r_{i_{\perp}^*, m_0}^{\text{LOT}} \xleftarrow{\$} \{0, 1\}^{\lambda} \quad \text{for } m_0 \in [M_0]. \\
& - \text{H}_3^b \text{ removes the unused labels from } \text{LOT.ct}_{i_{\perp}^*, m_0} \text{'s by setting} \\
& \quad \text{LOT.ct}_{i_{\perp}^*, m_0} \leftarrow \text{LOT.Send}(\text{hk}, h, (i_{\perp}^* - 1)M_0 + m_0, \\
& \quad \quad L_{i_{\perp}, m_0, \text{pk}_{i_{\perp}^*}^* [m_0]}, L_{i_{\perp}, m_0, \text{pk}_{i_{\perp}^*}^* [m_0]}; r_{i_{\perp}^*, m_0}^{\text{LOT}}) \quad \text{for } m_0 \in [M_0]. \\
& - \text{H}_4^b \text{ changes } \widehat{C}_{\text{ct}, i_{\perp}^*} \text{ into simulation, i.e.,}
\end{aligned}$$

$$\begin{aligned}
& \text{PKE.ct}_{i_{\perp}^*} \leftarrow \begin{cases} \text{PKE.Enc}(\text{pk}_{i_{\perp}^*}^*, \mu_{\perp}; r^{\text{PKE}}), & \text{if } b = 0; \\ \text{PKE.Enc}(\text{pk}_{i_{\perp}^*}^*, \mu_{\perp}; r^{\text{PKE}}), & \text{if } b = 1; \end{cases} \\
& (\widehat{C}_{\text{ct}, i_{\perp}^*}, \{L_{i_{\perp}, m_0, \text{pk}_{i_{\perp}^*}^* [m_0]}\}_{m_0 \in [M_0]}) \xleftarrow{\$} \text{GC.SimGarble}(C_{\text{ct}}, \text{pk}_{i_{\perp}^*}^*, \text{PKE.ct}_{i_{\perp}^*}),
\end{aligned}$$

where $\text{pk}_{i_{\perp}^*}^* = \text{pk}$ is sampled by the experiment (not adversarially controlled).

The following claims hold, all of which are immediate by inspection:

- Claim 5.* $\text{H}_0^b \approx \text{H}_1^b$ for $b \in \{0, 1\}$ if Obf is an $i\mathcal{O}$ for $\text{poly}(\lambda)$ -sized domain.
- Claim 6.* $\text{H}_1^b \approx \text{H}_2^b$ for $b \in \{0, 1\}$ if PPRF is pseudorandom at the punctured point.
- Claim 7.* $\text{H}_2^b \approx \text{H}_3^b$ for $b \in \{0, 1\}$ if LOT is database-selectively sender-private.
- Claim 8.* $\text{H}_3^b \approx \text{H}_4^b$ for $b \in \{0, 1\}$ if GC is w -hiding.
- Claim 9.* $\text{H}_4^0 \approx \text{H}_4^1$ if PKE is semantically secure for random messages.

$\text{Exp}_{\text{IH}}^0 \approx \text{Exp}_{\text{IH}}^1$ follows from a hybrid argument. \square

5 AH-BTR from AH-PLBE

Ingredient of Construction 2. Let $\text{ahPLBE} = (\text{ahPLBE.Gen}, \text{ahPLBE.Enc}, \text{ahPLBE.Dec})$ be an AH-PLBE scheme.

Construction 2 (adapted from [11; Sect. 2.2]). Our AH-BTR works as follows:

- Gen is the same as ahPLBE.Gen.
- Enc($\{\text{pk}_j\}_{j \in [N]}, \mu$) runs and outputs $\text{ct} \xleftarrow{\$} \text{ahPLBE.Enc}(\{\text{pk}_j\}_{j \in [N]}, 0, \mu)$.
- Dec is the same as ahPLBE.Dec.
- $\text{Trace}^{\mathcal{D}}(\{\text{pk}_j^*\}_{j \in [N]}, 1^{1/\varepsilon^*})$ defines for $i \in [0..N]$,

$$\varepsilon_i = \Pr \left[\underbrace{\begin{array}{l} \mu_0 \xleftarrow{\$} \{0, 1\}^\lambda, \quad \mu_1 \xleftarrow{\$} \{0, 1\}^\lambda, \quad \beta \xleftarrow{\$} \{0, 1\} \\ \text{ct} \xleftarrow{\$} \text{ahPLBE.Enc}(1^\lambda, \{\text{pk}_j^*\}_{j \in [N]}, i, \mu_\beta) \end{array}}_{\text{experiment } \mathcal{E}_i \text{ (sampling and testing) and event } E_i \text{ (correct guessing)}} : \mathcal{D}(\mu_0, \mu_1, \text{ct}) = \beta \right] - \frac{1}{2}.$$

Setting $\delta \leftarrow \frac{\varepsilon^*}{10N}$ and $\eta \leftarrow \left\lceil \frac{\lambda + \log(2N+2)}{2\delta^2} \right\rceil$, for each $i \in [0..N]$, the algorithm runs \mathcal{E}_i for η times independently, counts the absolute frequency $\xi_i \in [0..\eta]$ of E_i , and computes $\hat{\varepsilon}_i = \frac{\xi_i}{\eta} - \frac{1}{2}$. It outputs

$$i^* = \begin{cases} \min T, & \text{if } T \leftarrow \{i \in [N] : |\hat{\varepsilon}_i - \hat{\varepsilon}_{i-1}| \geq 3\delta\} \neq \emptyset; \\ \perp, & \text{if } T = \emptyset. \end{cases}$$

Robustness Correctness, Efficiency, Compatibility. These are inherited from the underlying AH-PLBE. When based on Construction 1, the resultant AH-BTR has

$$T_{\text{Enc}} = O(N) \text{ poly}(\lambda), \quad |\text{ct}| = \text{poly}(\lambda), \quad T_{\text{Dec}} = O(N) \text{ poly}(\lambda),$$

and is compatible with the existing public-key encryption schemes.

Theorem 10 (¶). *Suppose in Construction 2, the AH-PLBE scheme ahPLBE is message-hiding, then the resultant AH-BTR is complete.*

Theorem 11 (¶). *Suppose in Construction 2, the AH-PLBE scheme ahPLBE is index-hiding, then the resultant AH-BTR is sound.*

Proof (Theorem 10). Consider any efficient adversary \mathcal{C} against the completeness of Construction 2. Let **GoodEst** be the event that $|\hat{\varepsilon}_i - \varepsilon_i| \leq \delta$ for all $i \in [0..N]$. By the Chernoff bound, the union bound, and the law of total probability,

$$\Pr[\neg \text{GoodEst}] = \mathbb{E}[\Pr[\neg \text{GoodEst} \mid \varepsilon^*, N]] \leq \mathbb{E}[2(N+1) \exp(-2\delta^2 \eta)] \leq 2^{-\lambda}.$$

Let **BadEnd** be the event that $|\varepsilon_N| > \frac{\varepsilon^*}{2}$, then **GoodDist** \wedge \neg **BadEnd** implies

$$\begin{aligned} \max_{i \in [N]} |\varepsilon_{i-1} - \varepsilon_i| &\geq \frac{1}{N} \sum_{i=1}^N |\varepsilon_{i-1} - \varepsilon_i| \geq \frac{1}{N} \left| \sum_{i=1}^N (\varepsilon_{i-1} - \varepsilon_i) \right| = \frac{1}{N} |\varepsilon_0 - \varepsilon_N| \\ &\geq \frac{1}{N} (|\varepsilon_0| - |\varepsilon_N|) \underset{\substack{\uparrow \\ \text{GoodDist}}}{\geq} \frac{1}{N} \left(\varepsilon^* - \frac{\varepsilon^*}{2} \right) \underset{\substack{\uparrow \\ \neg \text{BadEnd}}}{\geq} \frac{\varepsilon^*}{2N} = 5\delta. \end{aligned}$$

Therefore, **GoodDist** \wedge \neg **BadEnd** \wedge **GoodEst** implies

$$\max_{i \in [N]} |\hat{\varepsilon}_{i-1} - \hat{\varepsilon}_i| \underset{\substack{\uparrow \\ \text{GoodEst}}}{\geq} \max_{i \in [N]} (|\varepsilon_{i-1} - \varepsilon_i| - 2\delta) \underset{\substack{\uparrow \\ \text{GoodDist} \wedge \neg \text{BadEnd}}}{\geq} 5\delta - 2\delta = 3\delta,$$

which in turn implies $T \neq \emptyset$ hence $i^* \in [N]$, i.e., $\neg \text{NotFound}$. By contraposition,

$$\text{GoodDist} \wedge \text{NotFound} \wedge \text{GoodEst} \implies \text{BadEnd}.$$

By the union bound,

$$\begin{aligned} \Pr[\mathcal{C} \text{ wins}] &\leq \Pr[\neg \text{GoodEst}] + \Pr[(\mathcal{C} \text{ wins}) \wedge \text{GoodEst}] \\ &= \Pr[\neg \text{GoodEst}] + \Pr[\text{GoodDist} \wedge \text{NotFound} \wedge \text{GoodEst}] \\ &\leq 2^{-\lambda} + \Pr[\text{BadEnd}], \end{aligned}$$

so it remains to show $\Pr[\text{BadEnd}] = \text{negl}(\lambda)$.

Consider the following efficient adversary \mathcal{A} against the message-hiding property of **ahPLBE**:

- \mathcal{A} runs \mathcal{C} to obtain

$$\mathcal{D}, \quad \{\text{pk}_j^*\}_{j \in [N]}, \quad 1^{1/\varepsilon^*}.$$

- \mathcal{A} runs \mathcal{E}_N once and notes down $\alpha \in \{0, 1\}$ indicating whether E_N happened, i.e., $\alpha = 1$ if and only if \mathcal{D} guessed correctly in the trial.
- \mathcal{A} submits $\{\text{pk}_j^*\}_{j \in [N]}$ to the message-hiding experiment, receives $(\mu_0, \mu_1, \text{ct})$ back, and runs and outputs $b' \xleftarrow{\$} \mathcal{D}(\mu_0, \mu_1, \text{ct}) \oplus \alpha$.

Routine calculation shows that the advantage of \mathcal{A} is $\mathbb{E}[4\varepsilon_N^2]$, which must be negligible by the message-hiding property of **ahPLBE**. Let $B = \text{poly}(\lambda)$ be an upper bound of $1/\varepsilon^*$ (B exists since \mathcal{C} outputs $1^{1/\varepsilon^*}$ in polynomial time). By Markov's inequality,

$$\begin{aligned} \Pr[\text{BadEnd}] &= \Pr[4\varepsilon_N^2 > (\varepsilon^*)^2] \leq \Pr[4\varepsilon_N^2 > B^{-2}] \\ &\leq B^2 \mathbb{E}[4\varepsilon_N^2] = (\text{poly}(\lambda))^2 \text{negl}(\lambda) = \text{negl}(\lambda). \quad \square \end{aligned}$$

Proof (Theorem 11). Consider any efficient adversary \mathcal{C} against the soundness of Construction 2. Similarly to the **proof** of Theorem 10, define **GoodEst** and recall that $\Pr[\neg \text{GoodEst}] \leq 2^{-\lambda}$. We have

$$\begin{aligned} \Pr[\mathcal{C} \text{ wins}] &\leq \Pr[\neg \text{GoodEst}] + \Pr[(\mathcal{C} \text{ wins}) \wedge \text{GoodEst}] \\ &= \Pr[\neg \text{GoodEst}] + \Pr[\text{FalsePos} \wedge \text{GoodEst}] \\ &\leq 2^{-\lambda} + \Pr[\text{FalsePos} \wedge \text{GoodEst}], \end{aligned}$$

and it suffices to prove $\Pr[\text{FalsePos} \wedge \text{GoodEst}] = \text{negl}(\lambda)$.

Let α be a random element in an execution of **Trace** with

$$\alpha = \begin{cases} 0, & \text{if } i^* \in [N] \text{ and } \widehat{e}_{i^*-1} - \widehat{e}_{i^*} \geq 3\delta; \\ 1, & \text{if } i^* \in [N] \text{ and } \widehat{e}_{i^*-1} - \widehat{e}_{i^*} \leq -3\delta; \\ \perp, & \text{if } i^* = \perp. \end{cases}$$

Consider the following efficient adversary \mathcal{A} against the index-hiding property of **ahPLBE**:

– $\mathcal{A}(\text{pk})$ runs $\mathcal{C}(\text{pk})$ to obtain

$$\mathcal{D}, \quad N, \quad i_{\perp}^*, \quad \{\text{pk}_j^*\}_{j \in [N] \setminus \{i_{\perp}^*\}}, \quad 1^{1/\varepsilon^*},$$

and sets $\text{pk}_{i_{\perp}^*}^* \leftarrow \text{pk}$.

– \mathcal{A} runs

$$i^* \xleftarrow{\$} \text{Trace}^{\mathcal{D}}(\{\text{pk}_j^*\}_{j \in [N]}, 1^{1/\varepsilon^*}),$$

and aborts if $i^* \neq i_{\perp}^*$.

– \mathcal{A} notes down $\alpha \in \{0, 1\}$ from the above execution of Trace , submits

$$N, \quad i_{\perp}^*, \quad \{\text{pk}_j^*\}_{j \in [N] \setminus \{i_{\perp}^*\}}$$

to the index-hiding experiment, gets (μ, ct) back, samples and sets

$$\beta \xleftarrow{\$} \{0, 1\}, \quad \mu_{\beta} \leftarrow \mu, \quad \mu_{\neg\beta} \xleftarrow{\$} \{0, 1\}^{\lambda},$$

and runs and outputs $b' \xleftarrow{\$} \mathcal{D}(\mu_0, \mu_1, \text{ct}) \oplus \neg\beta \oplus \alpha$.

Routine calculation shows that the advantage of \mathcal{A} is

$$\mathbb{E}[\mathbb{1}_{\text{FalsePos}} \cdot (-1)^{\alpha}(\varepsilon_{i^*-1} - \varepsilon_{i^*})],$$

which must be negligible by the index-hiding property of **ahPLBE**.

Let $B = \text{poly}(\lambda)$ be an upper bound of $10N/\varepsilon^*$ (B exists since \mathcal{C} outputs 1^N and $1^{1/\varepsilon^*}$ in polynomial time). The event $\text{FalsePos} \wedge \text{GoodEst}$ implies

$$\begin{aligned} |(\varepsilon_{i^*-1} - \varepsilon_{i^*}) - (\widehat{\varepsilon}_{i^*-1} - \widehat{\varepsilon}_{i^*})| &\leq 2\delta < 3\delta \leq |\widehat{\varepsilon}_{i^*-1} - \widehat{\varepsilon}_{i^*}| \\ \implies (-1)^{\alpha}(\varepsilon_{i^*-1} - \varepsilon_{i^*}) &= |\varepsilon_{i^*-1} - \varepsilon_{i^*}| \geq 3\delta - 2\delta = \frac{\varepsilon^*}{10N} \geq B^{-1}. \end{aligned}$$

Moreover, $(-1)^{\alpha}(\varepsilon_{i^*-1} - \varepsilon_{i^*}) \geq -1$ always holds. These together show that

$$\begin{aligned} &\Pr[\text{FalsePos} \wedge \text{GoodEst}] \\ &= B \mathbb{E}[\mathbb{1}_{\text{FalsePos}} \cdot \mathbb{1}_{\text{GoodEst}} \cdot B^{-1}] \\ &\leq B \mathbb{E}[\mathbb{1}_{\text{FalsePos}} \cdot \mathbb{1}_{\text{GoodEst}} \cdot (-1)^{\alpha}(\varepsilon_{i^*-1} - \varepsilon_{i^*})] \\ &\leq B \left(\mathbb{E}[\mathbb{1}_{\text{FalsePos}} \cdot \mathbb{1}_{\text{GoodEst}} \cdot (-1)^{\alpha}(\varepsilon_{i^*-1} - \varepsilon_{i^*})] \right. \\ &\quad \left. + \mathbb{E}[\mathbb{1}_{\text{FalsePos}} \cdot \mathbb{1}_{\neg\text{GoodEst}} \cdot (-1)^{\alpha}(\varepsilon_{i^*-1} - \varepsilon_{i^*})] + \mathbb{E}[\mathbb{1}_{\text{FalsePos}} \cdot \mathbb{1}_{\neg\text{GoodEst}}] \right) \\ &= B \left(\mathbb{E}[\mathbb{1}_{\text{FalsePos}} \cdot (-1)^{\alpha}(\varepsilon_{i^*-1} - \varepsilon_{i^*})] + \Pr[\text{FalsePos} \wedge \neg\text{GoodEst}] \right) \\ &\leq B \left(\mathbb{E}[\mathbb{1}_{\text{FalsePos}} \cdot (-1)^{\alpha}(\varepsilon_{i^*-1} - \varepsilon_{i^*})] + 2^{-\lambda} \right) \\ &= \text{poly}(\lambda)(\text{negl}(\lambda) + 2^{-\lambda}) = \text{negl}(\lambda). \end{aligned} \quad \square$$

6 Trading Ciphertext Size for Decryption Time

While Construction 2 achieves constant ciphertext size, it takes time $\Omega(N)$ to decrypt. In contrast, the naïve scheme that encrypts to each user separately has $\Omega(N)$ -size ciphertext, yet decryption only takes constant time. By grouping the recipients and encrypting to each group separately, we can trade ciphertext size for decryption time.¹³ Previous work [64] already systemizes the idea of grouping in the context of traditional traitor tracing.

Ingredients of Construction 3. Let $\text{old} = (\text{old.Gen}, \text{old.Enc}, \text{old.Dec}, \text{old.Trace})$ be an AH-BTR scheme and γ some¹⁴ constant ($0 < \gamma < 1$).

Construction 3 (adapted from [64; Theorem 1]). Our new AH-BTR works as follows:

- Gen is the same as old.Gen.
- $\text{Enc}(\{\text{pk}_j\}_{j \in [N]}, \mu)$ sets $N_1 = \lceil N^\gamma \rceil$ and $N_2 = \lceil N/N_1 \rceil$. It runs

$$\text{old.ct}_{j_1} \stackrel{\$}{\leftarrow} \text{old.Enc}(\{\text{pk}_j\}_{(j_1-1)N_2 < j \leq j_1 N_2}, \mu) \quad \text{for } j_1 \in [N_1].$$

The algorithm outputs $\text{ct} = \{\text{old.ct}_{j_1}\}_{j_1 \in [N_1]}$.

- $\text{Dec}^{\{\text{pk}_j\}_{j \in [N]}, \text{ct}}(N, i, \text{sk}_i)$ sets $N_1 = \lceil N^\gamma \rceil$ and $N_2 = \lceil N/N_1 \rceil$. It parses ct as $\{\text{old.ct}_{j_1}\}_{j_1 \in [N_1]}$, finds $i_1 \in [N_1]$ such that $(i_1 - 1)N_2 < i \leq i_1 N_2$, and sets $N'_2 = \min\{N_2, N - (i_1 - 1)N_2\}$. The algorithm runs and outputs

$$\text{old.Dec}^{\{\text{pk}_j\}_{(i_1-1)N_2 < j \leq i_1 N_2}, \text{old.ct}_{i_1}}(N'_2, i - (i_1 - 1)N_2, \text{sk}_i).$$

- $\text{Trace}^{\mathcal{D}}(\{\text{pk}_j^*\}_{j \in [N]}, 1^{1/\varepsilon^*})$ sets $N_1 = \lceil N^\gamma \rceil$ and $N_2 = \lceil N/N_1 \rceil$. It runs

$$i_{j_1}^* \stackrel{\$}{\leftarrow} \text{old.Trace}^{\mathcal{D}_{j_1}}(\{\text{pk}_j^*\}_{(j_1-1)N_2 < j \leq j_1 N_2}, 1^{N_1/\varepsilon^*}) \quad \text{for } j_1 \in [N_1],$$

where $\mathcal{D}_{j_1}(\mu_0, \mu_1, \text{old.ct}^*)$ runs and outputs $\mathcal{D}(\mu_0, \mu_1, \{\text{old.ct}_{j'_1}\}_{j'_1 \in [N_1]})$ with

$$\text{old.ct}_{j'_1} \begin{cases} \stackrel{\$}{\leftarrow} \text{old.Enc}(\{\text{pk}_j^*\}_{(j'_1-1)N_2 < j \leq j'_1 N_2}, \mu_0), & \text{if } j'_1 < j_1; \\ \leftarrow \text{old.ct}^*, & \text{if } j'_1 = j_1; \\ \stackrel{\$}{\leftarrow} \text{old.Enc}(\{\text{pk}_j^*\}_{(j'_1-1)N_2 < j \leq j'_1 N_2}, \mu_1), & \text{if } j'_1 > j_1. \end{cases}$$

The algorithm outputs

$$\begin{cases} (j_1 - 1)N_2 + i_{j_1}^*, & \text{if } i_{j'_1}^* = \perp \text{ for all } j'_1 < j_1 \text{ and } i_{j_1}^* \neq \perp; \\ \perp, & \text{if } i_{j'_1}^* = \perp \text{ for all } j'_1 \in [N_1]. \end{cases}$$

¹³ Alternatively, one can reformulate Construction 2 as a compiler that trades decryption time for ciphertext size, by grouping the recipients and compressing the groups. We refrained from such a formulation because the “transformation” uses a quite strong additional assumption, namely functional encryption for general circuits.

¹⁴ We require that $N \mapsto \lceil N^\gamma \rceil$ can be computed in (deterministic) time $\text{poly}(\log N)$.

Robustness Correctness and Compatibility. These are inherited from the underlying AH-BTR. When based on Construction 2, the resultant AH-BTR is compatible with the existing public-key encryption schemes.

Efficiency. Let $\gamma_1, \gamma_2, \gamma_3$ be constants such that the AH-BTR efficiency is

$$T_{\text{Enc}} = O(N^{\gamma_1}) \text{poly}(\lambda), \quad |\text{ct}| = O(N^{\gamma_2}) \text{poly}(\lambda), \quad T_{\text{Dec}} = O(N^{\gamma_3}) \text{poly}(\lambda),$$

then the underlying efficiency is mapped to the resultant efficiency¹⁵ by

$$(\gamma_1, \gamma_2, \gamma_3) \mapsto (1 - \gamma + \gamma\gamma_1, 1 - \gamma + \gamma\gamma_2, \gamma\gamma_3).$$

When based on Construction 2, the resultant AH-BTR enjoys

$$T_{\text{Enc}} = O(N) \text{poly}(\lambda), \quad |\text{ct}| = O(N^{1-\gamma}) \text{poly}(\lambda), \quad T_{\text{Dec}} = O(N^\gamma) \text{poly}(\lambda).$$

Theorem 12 (¶). Suppose in Construction 3, the underlying AH-BTR scheme old is complete, then so is the resultant AH-BTR.

Theorem 13 (¶). Suppose in Construction 3, the underlying AH-BTR scheme old is sound, then so is the resultant AH-BTR.

Proof (Theorem 12). Let \mathcal{C} be an efficient adversary against the completeness of the resultant scheme. Consider the following efficient adversary \mathcal{C}_{old} against the completeness of old:

- \mathcal{C}_{old} launches \mathcal{C} to obtain

$$\mathcal{D}, \quad \{\text{pk}_j^*\}_{j \in [N]}, \quad 1^{1/\varepsilon^*}.$$

It computes N_1, N_2 as specified by the resultant scheme.

- \mathcal{C}_{old} samples $j_1^* \xleftarrow{\$} [N_1]$, prepares $\mathcal{D}_{j_1^*}$ (using \mathcal{D} , as specified by the resultant scheme), and outputs

$$\mathcal{D}_{j_1^*}, \quad \{\text{pk}_j^*\}_{(j_1^*-1)N_2 < j \leq j_1^* N_2}, \quad 1^{N_1/\varepsilon^*}.$$

Let $B = \text{poly}(\lambda)$ be an upper bound of N_1 . Routine calculation shows

$$\Pr[\mathcal{C}_{\text{old}} \text{ wins}] \geq \frac{1}{B} \Pr[\mathcal{C} \text{ wins}],$$

hence by the completeness of old,

$$\Pr[\mathcal{C} \text{ wins}] \leq B \Pr[\mathcal{C}_{\text{old}} \text{ wins}] = \text{poly}(\lambda) \text{negl}(\lambda) = \text{negl}(\lambda). \quad \square$$

Proof (Theorem 13). Let \mathcal{C} be an efficient adversary against the soundness of the resultant scheme. Consider the following efficient adversary \mathcal{C}_{old} against the soundness of old:

¹⁵ We assume that old.ct's are of deterministic length so Dec knows the location of each particular old.ct. Alternatively, Enc can store a look-up table of their locations in ct.

- $\mathcal{C}_{\text{old}}(\text{pk})$ launches $\mathcal{C}(\text{pk})$ to obtain

$$\mathcal{D}, \quad N, \quad i_{\perp}^*, \quad \{\text{pk}_j^*\}_{j \in [N] \setminus \{i_{\perp}^*\}}, \quad 1^{1/\varepsilon^*}.$$

It computes N_1, N_2 as specified by the resultant scheme.

- \mathcal{C}_{old} computes $j_1^* = \lceil i_{\perp}^* / N_2 \rceil$ and outputs

$$\mathcal{D}_{j_1^*}, \quad \min\{N_2, N - (j_1^* - 1)N_2\}, \quad i_{\perp}^* - (j_1^* - 1)N_2, \\ \{\text{pk}_j^*\}_{(j_1^* - 1)N_2 < j \leq j_1^* N_2, j \neq i_{\perp}^*}, \quad 1^{N_1/\varepsilon^*}.$$

Routine calculation and the soundness of old yield

$$\Pr[\mathcal{C} \text{ wins}] \leq \Pr[\mathcal{C}_{\text{old}} \text{ wins}] = \text{negl}(\lambda). \quad \square$$

7 Lower Bound on Ciphertext Size and Decryption Time

In this section, we prove that for all secure AH-BTR,

$$|\text{ct}| \cdot T_{\text{Dec}} = \Omega(N),$$

and therefore, we have constructed all the optimal (ignoring $\text{poly}(\lambda)$ factors) AH-BTR schemes in this work, completely pinning down the Pareto front of its efficiency. In fact, we will show a related bound against a restricted kind of broadcast encryption,¹⁶ which can be implemented using AH-BTR in a straightforward manner.

The scheme is *restricted* in the sense that the users are paired and encryption only broadcasts to those sets for which there is precisely one recipient from each pair. The required security notion is also weaker — it does not consider collusion among multiple non-recipients nor adaptive attacks.

Definition 18 (restricted broadcast encryption and its security). A restricted broadcast encryption (BE) scheme (*for the purpose of this work*) consists of 3 efficient algorithms:

- $\text{Gen}(1^\lambda, 1^N)$ takes a length parameter as input. It outputs a master public key mpk and a list $\{\text{sk}_{j,s}\}_{j \in [N], s \in \{0,1\}}$ of secret keys.
- $\text{Enc}(1^\lambda, \text{mpk}, R, \mu)$ takes as input the master public key mpk , an N -bit string $R \in \{0,1\}^N$, and a message $\mu \in \{0,1\}^\lambda$. It outputs a ciphertext ct_R .
- $\text{Dec}^{\text{mpk}, i, r, \text{sk}_{i,r}, R, \text{ct}_R}(1^\lambda)$ is given random access to the master public key mpk , a secret key with its description $(i, r, \text{sk}_{i,r})$, a ciphertext with its attribute (R, ct_R) . It is supposed to recover μ if and only if $R[i] = r$.

¹⁶ The lower bound thus also applies to all mildly expressive attribute-based encryption schemes.

The scheme must be correct, i.e., for all $\lambda, N \in \mathbb{N}$, $R \in \{0, 1\}^N$, $i \in [N]$, $\mu \in \{0, 1\}^\lambda$,

$$\Pr \left[\begin{array}{l} (\text{mpk}, \{\text{sk}_{j,s}\}_{j \in [N], s \in \{0,1\}}) \xleftarrow{\$} \text{Gen}(1^\lambda, 1^N) \\ \text{ct}_R \xleftarrow{\$} \text{Enc}(1^\lambda, \text{mpk}, R, \mu) \\ : \quad \text{Dec}^{\text{mpk}, i, R[i], \text{sk}_{i, R[i]}, R, \text{ct}_R}(1^\lambda) = \mu \end{array} \right] = 1.$$

The scheme is 1-key secure for random challenge against uniform adversaries (or secure for the purpose of this work) if

$$\begin{aligned} & \{(1^\lambda, 1^N, \text{mpk}, R, i^*, \mu_0, \text{sk}_{i^*, \neg R[i^*]}, \boxed{\text{ct}_0})\} \\ & \approx \{(1^\lambda, 1^N, \text{mpk}, R, i^*, \mu_0, \text{sk}_{i^*, \neg R[i^*]}, \boxed{\text{ct}_1})\} \end{aligned}$$

with the components being

$$\begin{aligned} & (\text{mpk}, \{\text{sk}_{j,s}\}_{j \in [N], s \in \{0,1\}}) \xleftarrow{\$} \text{Gen}(1^\lambda, 1^N), \quad R \xleftarrow{\$} \{0, 1\}^N, \quad i^* \xleftarrow{\$} [N], \\ & \text{for } b \in \{0, 1\}, \quad \mu_b \xleftarrow{\$} \{0, 1\}^\lambda, \quad \text{ct}_b \xleftarrow{\$} \text{Enc}(1^\lambda, \text{mpk}, R, \mu_b). \end{aligned}$$

for all polynomially bounded $N = N(\lambda)$,¹⁷ where the computational indistinguishability only has to hold against uniform adversaries.

Theorem 14 (¶). For all secure restricted BE,

$$\max |\text{ct}| \cdot \max T_{\text{Dec}} \geq \frac{N}{1000}$$

for all polynomially bounded $N = N(\lambda)$ and sufficiently large λ , where ct runs through all possible ciphertexts and T_{Dec} the time to probe R and produce output by Dec , both for R of length N .

We remark that while the statement and the proof here apply to perfectly correct schemes with polynomial security, it is straight-forward to adapt them for schemes with sufficient (say, constant) gap between correctness and security.

To prove Theorem 14, we need the following lemma:

Lemma 15 (adapted from [60; Theorem 2]). For all $N, P \in \mathbb{N}$ with $1 \leq P \leq N$, distribution D supported over Z , function $F : Z \times \{0, 1\}^N \rightarrow \{0, 1\}^S$, there exists a function $G : Z \times \{0, 1\}^N \rightarrow \{0, 1, \perp\}^N$ such that

$$|\{j \in [N] : G(z, R)[j] \neq \perp\}| \leq P \quad \text{for all } z \in Z \text{ and } R \in \{0, 1\}^N$$

and for all¹⁸ oracle (randomized) algorithm \mathcal{B}^Y making at most T queries to Y ,

$$|\Pr [\mathcal{B}^R(z, F(z, R)) \rightarrow 1] - \Pr [\mathcal{B}^H(z, F(z, R)) \rightarrow 1]| \leq \sqrt{\frac{ST}{2P}},$$

¹⁷ N need not be a computable function of λ . This does not make the security definition “non-uniform”, as a standard guessing argument (with advantage sign correction) applies to an interactive formulation in which the uniform and efficient \mathcal{A} chooses N .

¹⁸ Here, \mathcal{B}^Y need not be efficient for the lemma to hold. The particular \mathcal{B}^Y used in this work is efficient.

where

$$R \xleftarrow{\$} \{0, 1\}^N, \quad z \xleftarrow{\$} D, \quad H[j] \begin{cases} = G(z, R)[j], & \text{if } G(z, R)[j] \neq \perp; \\ \xleftarrow{\$} \{0, 1\}, & \text{if } G(z, R)[j] = \perp. \end{cases}$$

Proof (Theorem 14). Define

$$S = 1 + \max |\text{ct}|, \quad T = 1 + \max \{\text{number of bits in } R \text{ probed by Dec}\}.$$

For $\lambda, N \geq 1$, it is necessary that $|\text{ct}| \geq 1$ because ct can encode any string μ of length λ , and that $\max T_{\text{Dec}} \geq T$ because Dec performs all the probes and, in addition, produces at least 1 bit of output. Therefore,

$$\max |\text{ct}| \cdot \max T_{\text{Dec}} \geq \frac{\max |\text{ct}| + 1}{2} \cdot \max T_{\text{Dec}} \geq \frac{ST}{2}.$$

It remains to prove $ST \geq \frac{2N}{1000}$ for sufficiently large λ . It suffices to consider the case when $N \geq 2$ and $ST \leq 2N$.

We prepare for Lemma 15. Let P be determined later, and

$$z = \left(\begin{array}{c} \mu, z_{\text{Enc}}, \text{mpk}, \\ \{\text{sk}_{j,s}\}_{j \in [N], s \in \{0,1\}} \end{array} \right) \sim D = \left\{ \begin{array}{c} \mu \xleftarrow{\$} \{0, 1\}^\lambda \\ z_{\text{Enc}} : \text{randomness for Enc} \\ (\text{mpk}, \{\text{sk}_{j,s}\}_{j \in [N], s \in \{0,1\}}) \xleftarrow{\$} \text{Gen}(1^N) \end{array} \right\},$$

$$F(z, R) = 0^{S-|\text{ct}|-1} \|1\| \text{ct}, \quad \text{where } \text{ct} \leftarrow \text{Enc}(\text{mpk}, R, \mu; z_{\text{Enc}}).$$

Let G be the function guaranteed by Lemma 15 and make $\mathcal{B}^Y(z, f)$ do the following:

- Sample $i^* \xleftarrow{\$} [N]$ and query $r^* \leftarrow Y[i^*]$.
- Read $\mu, \text{mpk}, \text{sk}_{i^*, r^*}$ from z . Read ct from f .
- Run $\mu' \xleftarrow{\$} \text{Dec}^{\text{mpk}, i^*, r^*, \text{sk}_{i^*, r^*}, Y, \text{ct}}()$.
- Output 1 if and only if $\mu = \mu'$.

Note that \mathcal{B} indeed makes at most T queries to Y , the first to obtain r^* and the rest to run Dec .

For $w \in \{1, 2, 3, 4, 5\}$, write p_w for $\Pr[\mathcal{B}^{Y_w}(z, f; i^*) \rightarrow 1]$, where

$$\begin{aligned} i^* &\xleftarrow{\$} [N], & Y_1 &= R, \\ Y_2[j] &\begin{cases} = G(z, F(z, R))[j], & \text{if } G(z, F(z, R))[j] \neq \perp; \\ \xleftarrow{\$} \{0, 1\}, & \text{if } G(z, F(z, R))[j] = \perp; \end{cases} \\ Y_3[j] &\begin{cases} = G(z, F(z, R))[j], & \text{if } j \neq i^* \text{ and } G(z, F(z, R))[j] \neq \perp; \\ \xleftarrow{\$} \{0, 1\}, & \text{if } j \neq i^* \text{ and } G(z, F(z, R))[j] = \perp; \\ \xleftarrow{\$} \{0, 1\}, & \text{if } j = i^*; \end{cases} \\ Y_4[j] &\begin{cases} = R[j], & \text{if } j \neq i^*; \\ \xleftarrow{\$} \{0, 1\}, & \text{if } j = i^*; \end{cases} & Y_5[j] &\begin{cases} = R[j], & \text{if } j \neq i^*; \\ = \neg R[i^*], & \text{if } j = i^*. \end{cases} \end{aligned}$$

By the correctness of the restricted BE scheme, $p_1 = 1$.

From Lemma 15,

$$|p_1 - p_2| \leq \sqrt{\frac{ST}{2P}}, \quad |p_4 - p_3| \leq \sqrt{\frac{ST}{2P}}.$$

Here, the second inequality is obtained by applying the lemma to

$$\mathcal{C}^Y(z, f) = \mathcal{B}^{Y'}(z, f; i^*), \quad \text{where } i^* \xleftarrow{\$} [N], \quad Y'[j] \begin{cases} = Y[j], & \text{if } j \neq i^*; \\ \xleftarrow{\$} \{0, 1\}, & \text{if } j = i^*. \end{cases}$$

Clearly, $|p_2 - p_3| \leq \frac{P}{N}$. Setting $P = \left\lceil \sqrt[3]{\frac{STN^2}{2}} \right\rceil$, we have

$$\begin{aligned} |p_1 - p_4| &\leq |p_1 - p_2| + |p_2 - p_3| + |p_3 - p_4| \\ &\leq \sqrt{\frac{ST}{2P}} + \frac{P}{N} + \sqrt{\frac{ST}{2P}} \leq 3\sqrt[3]{\frac{ST}{2N}} + \frac{1}{N} < 4\sqrt[3]{\frac{ST}{2N}}, \end{aligned}$$

where the last inequality follows from $N \geq 2$. By how $Y[i^*]$ is set,

$$p_4 = \frac{p_1 + p_5}{2} \implies p_5 = p_1 - 2(p_1 - p_4) \geq p_1 - 2|p_1 - p_4| > 1 - 8\sqrt[3]{\frac{ST}{2N}}.$$

Consider the following adversary $\mathcal{A}(\text{mpk}, R, i^*, \mu_0, \text{sk}_{i^*, \neg R[i^*]}, \text{ct})$ against the security of the restricted BE scheme:

- Construct Y_5 from R and let $r^* \leftarrow Y_5[i^*] = \neg R[i^*]$.
- Run $\mu' \xleftarrow{\$} \text{Dec}^{\text{mpk}, i^*, r^*, \text{sk}_{i^*, r^*}, Y_5, \text{ct}}()$, i.e., pretend $R[i^*]$ were $\neg R[i^*]$ and try decrypting using the key given to \mathcal{A} .
- Output 1 if and only if $\mu' = \mu_0$.

If $\text{ct} = \text{ct}_1$ is an encryption of μ_1 , then μ_0 is uniformly random and independent of everything else, hence

$$\Pr[\mathcal{A}(\dots) \rightarrow 1 \text{ with } \text{ct} = \text{ct}_1] \leq 2^{-\lambda}.$$

Note that \mathcal{A} is a uniform adversary. By the security of the restricted BE scheme,

$$p_5 = \Pr[\mathcal{B}^{Y_5}(z, f; i^*) \rightarrow 1] = \Pr[\mathcal{A}(\dots) \rightarrow 1 \text{ with } \text{ct} = \text{ct}_0] \leq 2^{-\lambda} + \text{negl}(\lambda) < \frac{1}{5}$$

for sufficiently large λ , which gives

$$1 - 8\sqrt[3]{\frac{ST}{2N}} < p_5 < \frac{1}{5} \implies ST > \frac{2N}{1000}. \quad \square$$

Corollary 16 (¶). *For all secure AH-BTR,*

$$\max |\text{ct}| \cdot \max T_{\text{Dec}} \geq \frac{N}{1000}$$

for all polynomially bounded $N = N(\lambda)$ and sufficiently large λ , where T_{Dec} only counts the time to probe pk_j 's and produce output. Ignoring $\text{poly}(\lambda)$ factors, Construction 3 achieves all possible optimal trade-offs in terms of the exponents over N in the dependency of ciphertext size and (actual) decryption time, fully demonstrating the Pareto front of AH-BTR efficiency.

Proof (Corollary 16). Let

$$\text{ahBTR} = (\text{ahBTR.Gen}, \text{ahBTR.Enc}, \text{ahBTR.Dec}, \text{ahBTR.Trace})$$

be a secure AH-BTR and construct the following restricted BE scheme:

- $\text{Gen}(1^N)$ runs

$$(\text{pk}_{j,s}, \text{sk}_{j,s}) \xleftarrow{\$} \text{ahBTR.Gen}() \quad \text{for } j \in [N], s \in \{0, 1\}$$

and outputs $\text{mpk} = \{\text{pk}_{j,s}\}_{j \in [N], s \in \{0, 1\}}$ with $\{\text{sk}_{j,s}\}_{j \in [N], s \in \{0, 1\}}$.

- $\text{Enc}(\text{mpk}, R, \mu)$ runs and outputs

$$\text{ct} \xleftarrow{\$} \text{ahBTR.Enc}(\{\text{pk}_{j,R[j]}\}_{j \in [N]}, \mu).$$

- $\text{Dec}^{\text{mpk}, i, r, \text{sk}_{i,r}, R, \text{ct}}()$ runs $\text{ahBTR.Dec}^{K, \text{ct}}(N, i, \text{sk}_{i,r})$, where K is an oracle implemented by Dec for ahBTR.Dec to probe pk_j 's. Whenever ahBTR.Dec probes $\text{pk}_j[m_0]$, we make Dec probe $R[j]$ and answer $\text{pk}_{j,R[j]}[m_0]$.

It is straight-forward to verify that the constructed scheme is correct and secure. Since a restricted BE ciphertext is precisely an AH-BTR ciphertext, each probe to pk_j 's by ahBTR.Dec translates to exactly one probe to $R[j]$ by Dec with no more additional probes by Dec on its own, and Dec outputs whatever ahBTR.Dec outputs, the corollary follows from Theorem 14. \square

Acknowledgement. The author was supported by NSF grants CNS-1936825 (CAREER), CNS-2026774, a J.P. Morgan AI Research Award, and a Simons Collaboration on the Theory of Algorithmic Fairness. The views expressed in this work are those of the author and do not reflect the official policy or position of any of the supporters. The author thanks Huijia Lin for her encouragement and support for him to pursue an independent research project (culminating this paper), Hoeteck Wee for helpful suggestions on writing from another context, and Daniel Wichs for helpful discussions. He started researching this topic after rereading a post [59] on V2EX.

References

1. Agrawal, S., Wichs, D., Yamada, S.: Optimal broadcast encryption from LWE and pairings in the standard model. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part I. LNCS, vol. 12550, pp. 149–178. Springer, Heidelberg (Nov 2020). https://doi.org/10.1007/978-3-030-64375-1_6

2. Agrawal, S., Yamada, S.: Optimal broadcast encryption from pairings and LWE. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part I. LNCS, vol. 12105, pp. 13–43. Springer, Heidelberg (May 2020). https://doi.org/10.1007/978-3-030-45721-1_2
3. Ananth, P., Lombardi, A.: Succinct garbling schemes from functional encryption through a local simulation paradigm. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, Part II. LNCS, vol. 11240, pp. 455–472. Springer, Heidelberg (Nov 2018). https://doi.org/10.1007/978-3-030-03810-6_17
4. Austrin, P., Kreitz, G.: Lower bounds for subset cover based broadcast encryption. In: Vaudenay, S. (ed.) AFRICACRYPT 08. LNCS, vol. 5023, pp. 343–356. Springer, Heidelberg (Jun 2008)
5. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (Aug 2001). https://doi.org/10.1007/3-540-44647-8_1
6. Bellare, M., Hoang, V.T., Rogaway, P.: Foundations of garbled circuits. In: Yu, T., Danezis, G., Gligor, V.D. (eds.) ACM CCS 2012. pp. 784–796. ACM Press (Oct 2012). <https://doi.org/10.1145/2382196.2382279>
7. Blundo, C., Cresti, A.: Space requirements for broadcast encryption. In: Santis, A.D. (ed.) EUROCRYPT’94. LNCS, vol. 950, pp. 287–298. Springer, Heidelberg (May 1995). <https://doi.org/10.1007/BFb0053444>
8. Boneh, D., Franklin, M.K.: An efficient public key traitor tracing scheme. In: Wiener, M.J. (ed.) CRYPTO’99. LNCS, vol. 1666, pp. 338–353. Springer, Heidelberg (Aug 1999). https://doi.org/10.1007/3-540-48405-1_22
9. Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (Aug 2005). https://doi.org/10.1007/11535218_16
10. Boneh, D., Naor, M.: Traitor tracing with constant size ciphertext. In: Ning, P., Syverson, P.F., Jha, S. (eds.) ACM CCS 2008. pp. 501–510. ACM Press (Oct 2008). <https://doi.org/10.1145/1455770.1455834>
11. Boneh, D., Sahai, A., Waters, B.: Fully collusion resistant traitor tracing with short ciphertexts and private keys. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 573–592. Springer, Heidelberg (May / Jun 2006). https://doi.org/10.1007/11761679_34
12. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (Mar 2011). https://doi.org/10.1007/978-3-642-19571-6_16
13. Boneh, D., Waters, B.: A fully collusion resistant broadcast, trace, and revoke system. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM CCS 2006. pp. 211–220. ACM Press (Oct / Nov 2006). <https://doi.org/10.1145/1180405.1180432>
14. Boneh, D., Waters, B.: Constrained pseudorandom functions and their applications. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 280–300. Springer, Heidelberg (Dec 2013). https://doi.org/10.1007/978-3-642-42045-0_15
15. Boneh, D., Waters, B., Zhandry, M.: Low overhead broadcast encryption from multilinear maps. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 206–223. Springer, Heidelberg (Aug 2014). https://doi.org/10.1007/978-3-662-44371-2_12

16. Boneh, D., Zhandry, M.: Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 480–499. Springer, Heidelberg (Aug 2014). https://doi.org/10.1007/978-3-662-44371-2_27
17. Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 501–519. Springer, Heidelberg (Mar 2014). https://doi.org/10.1007/978-3-642-54631-0_29
18. Brakerski, Z., Vaikuntanathan, V.: Lattice-inspired broadcast encryption and succinct ciphertext-policy ABE. Cryptology ePrint Archive, Report 2020/191 (2020), <https://eprint.iacr.org/2020/191>
19. Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (Feb 2007). https://doi.org/10.1007/978-3-540-70936-7_28
20. Chen, Y., Vaikuntanathan, V., Waters, B., Wee, H., Wichs, D.: Traitor-tracing from LWE made simple and attribute-based. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, Part II. LNCS, vol. 11240, pp. 341–369. Springer, Heidelberg (Nov 2018). https://doi.org/10.1007/978-3-030-03810-6_13
21. Cho, C., Döttling, N., Garg, S., Gupta, D., Miao, P., Polychroniadou, A.: Labeled oblivious transfer and its applications. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part II. LNCS, vol. 10402, pp. 33–65. Springer, Heidelberg (Aug 2017). https://doi.org/10.1007/978-3-319-63715-0_2
22. Chor, B., Fiat, A., Naor, M.: Tracing traitors. In: Desmedt, Y. (ed.) CRYPTO’94. LNCS, vol. 839, pp. 257–270. Springer, Heidelberg (Aug 1994). https://doi.org/10.1007/3-540-48658-5_25
23. Corrigan-Gibbs, H., Henzinger, A., Kogan, D.: Single-server private information retrieval with sublinear amortized time. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 3–33. Springer, Heidelberg (May / Jun 2022). https://doi.org/10.1007/978-3-031-07085-3_1
24. Damgård, I.B., Larsen, K.G., Yakubov, S.: Broadcast secret-sharing, bounds and applications. In: Tessaro, S. (ed.) 2nd Conference on Information-Theoretic Cryptography (ITC 2021). Leibniz International Proceedings in Informatics (LIPIcs), vol. 199, pp. 10:1–10:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2021). <https://doi.org/10.4230/LIPIcs.ITC.2021.10>
25. Daza, V., Herranz, J., Morillo, P., Ràfols, C.: *Ad-hoc* threshold broadcast encryption with shorter ciphertexts. Electronic Notes in Theoretical Computer Science **192**(2), 3–15 (2008). <https://doi.org/10.1016/j.entcs.2008.05.002>, proceedings of the Third Workshop on Cryptography for Ad-hoc Networks (WCAN 2007)
26. Delerablée, C.: Identity-based broadcast encryption with constant size ciphertexts and private keys. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 200–215. Springer, Heidelberg (Dec 2007). https://doi.org/10.1007/978-3-540-76900-2_12
27. Delerablée, C., Paillier, P., Pointcheval, D.: Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) PAIRING 2007. LNCS, vol. 4575, pp. 39–59. Springer, Heidelberg (Jul 2007). https://doi.org/10.1007/978-3-540-73489-5_4
28. Fiat, A., Naor, M.: Broadcast encryption. In: Stinson, D.R. (ed.) CRYPTO’93. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (Aug 1994). https://doi.org/10.1007/3-540-48329-2_40

29. Garg, S., Gentry, C., Sahai, A., Waters, B.: Witness encryption and its applications. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th ACM STOC. pp. 467–476. ACM Press (Jun 2013). <https://doi.org/10.1145/2488608.2488667>
30. Gay, R., Kerenidis, I., Wee, H.: Communication complexity of conditional disclosure of secrets and attribute-based encryption. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 485–502. Springer, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-48000-7_24
31. Gentry, C., Waters, B.: Adaptive security in broadcast encryption systems (with short ciphertexts). In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 171–188. Springer, Heidelberg (Apr 2009). https://doi.org/10.1007/978-3-642-01001-9_10
32. Gentry, C., Wichs, D.: Separating succinct non-interactive arguments from all falsifiable assumptions. In: Fortnow, L., Vadhan, S.P. (eds.) 43rd ACM STOC. pp. 99–108. ACM Press (Jun 2011). <https://doi.org/10.1145/1993636.1993651>
33. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions (extended abstract). In: 25th FOCS. pp. 464–479. IEEE Computer Society Press (Oct 1984). <https://doi.org/10.1109/SFCS.1984.715949>
34. Goldwasser, S., Kalai, Y.T.: Cryptographic assumptions: A position paper. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016-A, Part I. LNCS, vol. 9562, pp. 505–522. Springer, Heidelberg (Jan 2016). https://doi.org/10.1007/978-3-662-49096-9_21
35. Goyal, R., Koppula, V., Russell, A., Waters, B.: Risky traitor tracing and new differential privacy negative results. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 467–497. Springer, Heidelberg (Aug 2018). https://doi.org/10.1007/978-3-319-96884-1_16
36. Goyal, R., Koppula, V., Waters, B.: Collusion resistant traitor tracing from learning with errors. In: Diakonikolas, I., Kempe, D., Henzinger, M. (eds.) 50th ACM STOC. pp. 660–670. ACM Press (Jun 2018). <https://doi.org/10.1145/3188745.3188844>
37. Goyal, R., Koppula, V., Waters, B.: New approaches to traitor tracing with embedded identities. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019, Part II. LNCS, vol. 11892, pp. 149–179. Springer, Heidelberg (Dec 2019). https://doi.org/10.1007/978-3-030-36033-7_6
38. Goyal, R., Quach, W., Waters, B., Wichs, D.: Broadcast and trace with N^ϵ ciphertext size from standard assumptions. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 826–855. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-26954-8_27
39. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM CCS 2006. pp. 89–98. ACM Press (Oct / Nov 2006). <https://doi.org/10.1145/1180405.1180418>, available as Cryptology ePrint Archive Report 2006/309
40. Ishai, Y., Wee, H.: Partial garbling schemes and their applications. In: Esparza, J., Fraigniaud, P., Husfeldt, T., Koutsoupias, E. (eds.) ICALP 2014, Part I. LNCS, vol. 8572, pp. 650–662. Springer, Heidelberg (Jul 2014). https://doi.org/10.1007/978-3-662-43948-7_54
41. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from well-founded assumptions. In: Khuller, S., Williams, V.V. (eds.) 53rd ACM STOC. pp. 60–73. ACM Press (Jun 2021). <https://doi.org/10.1145/3406325.3451093>
42. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from LPN over \mathbb{F}_p , DLIN, and PRGs in NC^0 . In: Dunkelman, O., Dziembowski, S. (eds.) EU-

- ROCRYPT 2022, Part I. LNCS, vol. 13275, pp. 670–699. Springer, Heidelberg (May / Jun 2022). https://doi.org/10.1007/978-3-031-06944-4_23
43. Katz, J., Yerukhimovich, A.: On black-box constructions of predicate encryption from trapdoor permutations. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 197–213. Springer, Heidelberg (Dec 2009). https://doi.org/10.1007/978-3-642-10366-7_12
 44. Kiayias, A., Papadopoulos, S., Triandopoulos, N., Zacharias, T.: Delegatable pseudorandom functions and applications. In: Sadeghi, A.R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013. pp. 669–684. ACM Press (Nov 2013). <https://doi.org/10.1145/2508859.2516668>
 45. Kiayias, A., Yung, M.: On crafty pirates and foxy tracers. In: ACM Workshop on Security and Privacy in Digital Rights Management. pp. 22–39. DRM '01, Springer-Verlag, Berlin, Heidelberg (2001)
 46. Kitagawa, F., Nishimaki, R., Tanaka, K., Yamakawa, T.: Adaptively secure and succinct functional encryption: Improving security and efficiency, simultaneously. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 521–551. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-26954-8_17
 47. Kurosawa, K., Yoshida, T., Desmedt, Y., Burmester, M.: Some bounds and a construction for secure broadcast encryption. In: Ohta, K., Pei, D. (eds.) ASIACRYPT'98. LNCS, vol. 1514, pp. 420–433. Springer, Heidelberg (Oct 1998). https://doi.org/10.1007/3-540-49649-1_33
 48. Lin, H., Tessaro, S.: Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 630–660. Springer, Heidelberg (Aug 2017). https://doi.org/10.1007/978-3-319-63688-7_21
 49. Lindell, Y., Pinkas, B.: A proof of security of Yao's protocol for two-party computation. *Journal of Cryptology* **22**(2), 161–188 (Apr 2009). <https://doi.org/10.1007/s00145-008-9036-8>
 50. Liu, Q., Zhandry, M.: Decomposable obfuscation: A framework for building applications of obfuscation from polynomial hardness. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part I. LNCS, vol. 10677, pp. 138–169. Springer, Heidelberg (Nov 2017). https://doi.org/10.1007/978-3-319-70500-2_6
 51. Luby, M., Staddon, J.: Combinatorial bounds for broadcast encryption. In: Nyberg, K. (ed.) EUROCRYPT'98. LNCS, vol. 1403, pp. 512–526. Springer, Heidelberg (May / Jun 1998). <https://doi.org/10.1007/BFb0054150>
 52. Naor, D., Naor, M., Lotspiech, J.: Revocation and tracing schemes for stateless receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (Aug 2001). https://doi.org/10.1007/3-540-44647-8_3
 53. Naor, M., Pinkas, B.: Threshold traitor tracing. In: Krawczyk, H. (ed.) CRYPTO'98. LNCS, vol. 1462, pp. 502–517. Springer, Heidelberg (Aug 1998). <https://doi.org/10.1007/BFb0055750>
 54. Naor, M., Pinkas, B.: Efficient trace and revoke schemes. In: Frankel, Y. (ed.) FC 2000. LNCS, vol. 1962, pp. 1–20. Springer, Heidelberg (Feb 2001)
 55. Nishimaki, R., Wichs, D., Zhandry, M.: Anonymous traitor tracing: How to embed arbitrary information in a key. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 388–419. Springer, Heidelberg (May 2016). https://doi.org/10.1007/978-3-662-49896-5_14
 56. Phan, D.H., Pointcheval, D., Strefer, M.: Decentralized dynamic broadcast encryption. In: Visconti, I., Prisco, R.D. (eds.) SCN 12. LNCS, vol. 7485, pp. 166–183. Springer, Heidelberg (Sep 2012). https://doi.org/10.1007/978-3-642-32928-9_10

57. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys, D.B. (ed.) 46th ACM STOC. pp. 475–484. ACM Press (May / Jun 2014). <https://doi.org/10.1145/2591796.2591825>
58. Sahai, A., Waters, B.R.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (May 2005). https://doi.org/10.1007/11426639_27
59. sillydaddy: GPG file encryption: One encrypted file can be decrypted by many keys. <https://v2ex.com/t/759538> (Mar 2021), retrieved on 20 May 2022, archived at <https://web.archive.org/web/20220520040245/https://v2ex.com/t/759538>.
60. Unruh, D.: Random oracles and auxiliary input. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 205–223. Springer, Heidelberg (Aug 2007). https://doi.org/10.1007/978-3-540-74143-5_12
61. Wu, Q., Qin, B., Zhang, L., Domingo-Ferrer, J.: Ad hoc broadcast encryption (poster presentation). In: Al-Shaer, E., Keromytis, A.D., Shmatikov, V. (eds.) ACM CCS 2010. pp. 741–743. ACM Press (Oct 2010). <https://doi.org/10.1145/1866307.1866416>
62. Yao, A.C.C.: How to generate and exchange secrets (extended abstract). In: 27th FOCS. pp. 162–167. IEEE Computer Society Press (Oct 1986). <https://doi.org/10.1109/SFCS.1986.25>
63. Yao, A.C.C.: Coherent functions and program checkers (extended abstract). In: 22nd ACM STOC. pp. 84–94. ACM Press (May 1990). <https://doi.org/10.1145/100216.100226>
64. Zhandry, M.: New techniques for traitor tracing: Size $N^{1/3}$ and more from pairings. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part I. LNCS, vol. 12170, pp. 652–682. Springer, Heidelberg (Aug 2020). https://doi.org/10.1007/978-3-030-56784-2_22
65. Zhandry, M.: New techniques for traitor tracing: Size $N^{1/3}$ and more from pairings. Cryptology ePrint Archive, Report 2020/954 (2020), <https://eprint.iacr.org/2020/954>
66. Zhandry, M.: Schrödinger’s pirate: How to trace a quantum decoder. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part III. LNCS, vol. 12552, pp. 61–91. Springer, Heidelberg (Nov 2020). https://doi.org/10.1007/978-3-030-64381-2_3
67. Zhandry, M.: White box traitor tracing. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part IV. LNCS, vol. 12828, pp. 303–333. Springer, Heidelberg, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84259-8_11