Broadcast, Trace and Revoke with Optimal Parameters from Polynomial Hardness

Shweta Agrawal[1], Simran Kumari[1], Anshu Yadav[1], and Shota Yamada[2]

[1] IIT Madras, Chennai, India
shweta@cse.iitm.ac.in, sim78608@gmail.com,
anshu.yadav06@gmail.com
[2] AIST Tokyo, Japan
yamada-shota@aist.go.jp

Abstract. A *broadcast, trace and revoke* system generalizes broadcast encryption as well as traitor tracing. In such a scheme, an encryptor can specify a list $L \subseteq N$ of revoked users so that (i) users in L can no longer decrypt ciphertexts, (ii) ciphertext size is independent of L, (iii) a pirate decryption box supports tracing of compromised users. The "holy grail" of this line of work is a construction which resists unbounded collusions, achieves all parameters (including public and secret key) sizes independent of $|L|$ and $|N|$, and is based on polynomial hardness assumptions. In this work we make the following contributions:

1. *Public Trace Setting:* We provide a construction which (i) achieves optimal parameters, (ii) supports embedding identities (from an exponential space) in user secret keys, (iii) relies on polynomial hardness assumptions, namely compact functional encryption (FE) and a key-policy attribute based encryption (ABE) with special efficiency properties, and (iv) enjoys adaptive security with respect to the revocation list. The previous best known construction by Nishimaki, Wichs and Zhandry (Eurocrypt 2016) which achieved optimal parameters and embedded identities, relied on indistinguishability obfuscation, which is considered an inherently subexponential assumption and achieved only selective security with respect to the revocation list.
2. *Secret Trace Setting:* We provide the first construction with optimal ciphertext, public and secret key sizes and embedded identities from any assumption outside Obfustopia. In detail, our construction relies on Lockable Obfuscation which can be constructed using LWE (Goyal, Koppula, Waters and Wichs, Zirdelis, Focs 2017) and two ABE schemes: (i) the key-policy scheme with special efficiency properties by Boneh et al. (Eurocrypt 2014) and (ii) a ciphertext-policy ABE for P which was recently constructed by Wee (Eurocrypt 2022) using a new assumption called *evasive and tensor* LWE. This assumption, introduced to build an ABE, is believed to be much weaker than lattice based assumptions underlying FE or iO – in particular it is required even for lattice based broadcast, without trace.

   Moreover, by relying on subexponential security of LWE, both our constructions can also support a *super-polynomial* sized revocation list,

so long as it allows efficient representation and membership testing. Ours is the first work to achieve this, to the best of our knowledge.

Keywords: Broadcast, Trace and Revoke · Revocable Predicate Encryption · Revocable Mixed Functional Encryption · Optimal Parameters

1 Introduction

Traitor Tracing. Traitor tracing (TT) schemes were first proposed by Chor, Fiat, and Naor [23] to enable content providers to trace malicious users who exploit their secret keys to construct illegal decryption boxes. More formally, a TT system is a public key encryption system comprising N users for some large polynomial N. Each user $i \in [N]$ is provided with a unique secret key sk_i for decryption, and there is a common public key pk which is used by the content distributor to encrypt content. If any collection of users attempts to create and sell a new decoding box that can be used to decrypt the content, then the tracing algorithm, given black-box access to any such pirate decoder, is guaranteed to output an index $i \in [N]$ of one of the corrupt users, which in turn allows to hold them accountable. The literature has considered both public and secret tracing, where the former requires knowledge of a secret key to run the trace procedure and the latter does not suffer from this restriction.

Broadcast Encryption. Broadcast Encryption [26] (BE) introduced by Fiat and Naor, is also an N user system which supports an encrypted broadcast functionality. In BE, a content provider can transmit a single ciphertext over a broadcast channel so that only an authorized subset $S \subseteq N$ of users can decrypt and recover the message. More formally, each user $i \in [N]$ is provided with a unique decryption key sk_i and a ciphertext ct_m for a message m also encodes an authorized list S so that sk_i decrypts ct_m if and only if $i \in S$. Evidently, public key encryption provides a trivial construction of BE with ciphertext of size $O(N)$ – thus, the focus in such schemes is to obtain short ciphertext, ideally logarithmic in N.

Broadcast, Trace and Revoke. Naor and Pinkas [43] suggested a meaningful interleaving of these two functionalities so that traitors that are identified by the TT scheme can be removed from the set of authorized users in a BE scheme. To capture this, they defined the notion of "Broadcast, Trace and Revoke" (or simply "Trace and Revoke", which we denote by TR) where the content provider in a broadcast encryption scheme includes a list L of revoked users in the ciphertext, and sk_i works to decrypt ct_L if $i \notin L$. Moreover, it is required that revocation remain compatible with tracing, so that if an adversary builds a pirate decoder that can decrypt ciphertexts encrypted with respect to L, then the tracing algorithm should be able to output a corrupt non-revoked user who participated in building the illegal decoder. Trace and revoke systems provide a functionality which is richer than a union of BE and TT, since the traitor traced by the latter must belong to the set of non-revoked users for the guarantee to

be meaningful. As such, TT schemes have been challenging to construct even given TT and BE schemes.

The Quest for Optimal Parameters. All the above primitives have been researched extensively over decades, resulting in a long sequence of beautiful constructions, non-exhaustively [14, 17, 15, 16, 44, 40, 33, 34, 6, 48]. A central theme in this line of work is to achieve optimal parameters, namely optimal sizes for the ciphertext, public key and secret key (and understanding tradeoffs thereof), while still supporting unbounded collusion resistance. Towards this, the powerful hammer of indistinguishability obfuscation (iO) [10] yielded the first feasibility results for traitor tracing [16] as well as trace and revoke [44] while multilinear maps [27, 24] led to the first construction for broadcast encryption [15]. Though there has been remarkable progress in the construction of iO from standard assumptions, with the breakthrough work of Jain, Lin and Sahai [38, 39] finally reaching this goal, iO is an inherently subexponential assumption [29] because the challenger is required to check whether two circuits are functionally equivalent, which can take exponential time in general. Indeed, all known constructions of iO assume subexponential hardness of the underlying algebraic assumptions. To address this limitation, a sequence of works [29, 3, 18, 28, 41] has sought to replace iO by polynomially hard assumptions such as functional encryption in different applications.

Optimal TT, BE and TR from Polynomial Assumptions: For traitor tracing, the first construction from standard assumptions was finally achieved by the seminal work of Goyal, Koppula and Waters [33] in the secret trace setting, from the Learning With Errors (LWE) assumption. For broadcast encryption, this goal was achieved by Agrawal and Yamada [6] from LWE and the bilinear GGM. In the standard model, Agrawal, Wichs and Yamada [5] provided a construction from a non-standard knowledge assumption on pairings, while Wee [48] provided a construction from a new assumption on lattices, called Evasive and Tensor LWE. For trace and revoke, the only construction without iO that achieves collusion resistance and optimal parameters is by Goyal, Vusirikala and Waters (GVW) [36] from *positional witness encryption* (PWE) which is a polynomial hardness assumption. However, their construction incurs an exponential loss in the security proof, requiring the underlying PWE to satisfy subexponential security. Moreover, although PWE is not an inherently subexponential assumption as are iO and witness encryption (WE), we do not currently know of any *constructions* of PWE that rely on standard polynomial hardness assumptions. In particular, [38, 39] do *not* imply PWE from polynomial hardness.

Pathway via Secret Tracing. Both the iO and PWE based constructions of TR [44, 36] achieve public tracing. Taking a lesson from TT, where optimal parameters were achieved from standard assumptions only in the secret trace setting [33], a natural approach towards optimal TR from better assumptions is to weaken the tracing algorithm to be secret key. This approach has been explored in a number of works – the current best parameters are achieved by Zhandry [51] who obtains the best known tradeoff in ciphertext, public key and secret key size.

In particular, Zhandry [51] showed that all parameters can be of size $O(N^{1/3})$ by relying on the bilinear generic group model (GGM). Note that the generic group model is a strong assumption, and indeed a construction secure in this model cannot be considered as relying on standard assumptions, since several non-standard assumptions on pairings are secure in the GGM. Prior to [51], Goyal et al. [35] provided a construction from LWE and Pairings, but their overall parameters are significantly worse – while their ciphertext can be arbitrarily small, $O(N^{\epsilon})$, their public key is $O(N)$ and secret key is $O(N^{c})$ for some large constant c[3].

Thus, a central open question in TR is:

Can we construct collusion resistant Trace and Revoke with optimal parameters from concrete polynomial assumptions?

Embedding Identities. Traditionally, it was assumed that tracing the index $i \in [N]$ of a corrupt user is enough, and there is an external mapping, maintained by the content distributor or some other party which associates the number i to the identity of the user, i.e. name, national identity number and such, which is then used to ensure accountability. The work of Nishimaki, Wichs and Zhandry (henceforth NWZ) [44] argued that this assumption is problematic since it implies that a user must trust the content provider with her confidential information. Storing such a map is particularly worrisome in the setting of public tracing since the user either cannot map the recovered index to an actual person, or the index-identity map must be stored publicly.

NWZ provided an appealing solution to the above conundrum – they suggest that identifying information be embedded in the key of the user, so that if a coalition of traitors constructs a pirate decoder, the tracing algorithm can directly retrieve the identifying information from one of the keys that was used to construct the decoder and no one needs to keep any records associating users to indices. Notably, the identities can live in an *exponential* sized space, which introduces significant challenges in the tracing procedure. Indeed, handling an exponential space in the tracing procedure is the key contribution of NWZ. They also provided constructions of traitor tracing as well as trace and revoke with embedded identities, denoted by EITT and EITR respectively, from various assumptions.

1.1 Prior Work: Embedded Identity Trace and Revoke.

In the *public* trace setting, the only work that achieves embedded identity trace and revoke (EITR) with full collusion resistance is that of NWZ. However, while it takes an important first step, the construction by NWZ suffers from the following drawbacks:

[3] Zhandry [51] states that the secret size in [35] is $O(N^2)$ but in fact the exponent is much larger due to the usage of arithmetic computations in NC^1, which blows up the circuit size associated with the ABE secret keys.

1. *Reliance on Subexponential Hardness Assumption.* The construction relies on indistinguishability obfuscation [10], which appears to be an inherently subexponential assumption as discussed above.
2. *Selective Security in Revocation List:* Despite relying on adaptive security of functional encryption, the notion of security achieved by their construction is selective – the adversary must announce the revocation list before making any key requests or seeing the challenge ciphertext.

In the *secret* trace setting, the work of Kim and Wu [40] achieves EITR from the subexponential Learning With Errors (LWE) assumption. However, their construction incurs a ciphertext size that grows with the size of the revocation list. Additionally, while they can achieve adaptive security with respect to the revocation list, this is either by incurring an exponential loss in the security proof, or by assuming sub-exponential security for an ingredient scheme.

1.2 Our Results

In this work, we provide the first constructions with optimal parameters from polynomial assumptions, which additionally support embedded identities from an exponential space. We detail our contributions below.

Public Trace Setting. We provide a construction of Trace and Revoke with public tracing which overcomes the limitations of NWZ – (i) it relies on polynomial hardness assumptions, namely functional encryption and "special" attribute based encryption, both of which can be constructed using standard polynomial hardness assumptions [13, 38, 39] (ii) it enjoys adaptive security in the revocation list.

A detailed comparison with prior work is provided in Table 1.

Work	|CT|	|SK|	|PK|	Trace Space	Sel/Adp	Asspn	Identities
[44]	1	1	1	Exp	Selective	Subexp (iO)	Yes
[36]	1	1	1	Poly	Adaptive	Subexp (subexp PWE)	No
This	1	1	1	Exp	Adaptive	Poly (FE and Special ABE)	Yes

Table 1. State of the art with Public Traceability.

Our Assumptions. Functional Encryption (FE) and Attribute Based Encryption (ABE) are generalizations of Public Key Encryption. In FE, a secret key corresponds to a circuit C and a ciphertext corresponds to an input x from the domain of C. Given a function key sk_C and a ciphertext ct_x, the decryptor can learn $C(x)$ and nothing else. It has been shown that FE implies iO [8, 12] albeit with exponential loss. The aforementioned work of Jain, Lin and Sahai [38, 39] provides a construction of compact FE from polynomial hardness assumptions, namely LPN, PRG in NC_0 and pairings. ABE is a special case of FE in which the input can be divided into a public and private part (x, m) and the circuit C in the secret key sk_C is only evaluated on the public part x in the ciphertext $\mathsf{ct}_{x,m}$. The private message m is revealed by decryption if and only if $C(x) = 1$. While FE implies ABE in general, we require our underlying ABE to satisfy special efficiency properties, which is not generically implied by FE. However, the desired ABE can be instantiated using the construction of Boneh et al. [13] which is based on LWE.

Secret Trace Setting. In the secret trace setting, we achieve the optimal size of $O(\log N)$ for ciphertext, public and secret key by relying on Lockable Obfuscation (LO) [32, 50] and two special ABE schemes – one, the key-policy scheme with special efficiency properties by Boneh et al. [13] which is based on LWE, and two, a ciphertext-policy ABE for P which was recently constructed by Wee [48] using the new evasive and tensor LWE assumptions. Along the way, we show that a small modification to the TR construction by Goyal et al. [35] yields a ciphertext of size $O(\log N)$ as against their original $O(N^\epsilon)$, from LWE and pairings. However, this construction retains the large public and secret keys of their construction, which depend at least linearly on N. Our results are summarized in Table 2.

Our Assumptions. We remark that while FE has now been constructed from standard assumptions [38, 39], the reliance of these constructions on pairings makes it insecure in the post-quantum regime. From lattices, constructions of FE rely on strong, non-standard assumptions which are often subject to attack [1, 4, 49, 30, 25, 37]. Hence, there is an active effort in the community [48, 46, 47] to construct advanced primitives from the hardness of weaker assumptions in the lattice regime. The new assumptions by Wee, also independently discovered by Tsabary [46], are formulated for designing ciphertext-policy ABE which is much weaker than FE since ABE is an all or nothing primitive in contrast to FE. As such, these are believed to be much weaker than lattice based assumptions that have been introduced in the context of FE or iO. In particular, based on the current state of art, evasive LWE is required even for broadcast encryption in the lattice regime, and is therefore necessary for the generalization of broadcast encryption studied in this work.

Super-polynomial Revoke List. Lastly, by relying on subexponential security of LWE, both our constructions can support a *super-polynomial* sized revocation list, so long as it allows efficient representation and membership testing. Ours is the first work to achieve this, to the best of our knowledge.

Work	$\|\mathsf{CT}\|$	$\|\mathsf{SK}\|$	$\|\mathsf{PK}\|$	Trace Space	Asspn	Identities
[35]	N^{ϵ}	N^{poly}	N	Poly	LWE and Pairings	No
[51]	N^{a}	N^{1-a}	N^{1-a}	Poly	GGM Pairings	No
[40]	L	1	1	Exp	Subexp LWE	Yes
This	1	1	1	Exp	Evasive Tensor LWE	Yes
Modified [35]	1	N^{c}	N	Poly	LWE and Pairings	No

Table 2. State of the Art with Secret Traceability. The column $|\mathsf{CT}|$ captures the dependence of ciphertext size on N and L where N denotes the number of users and L denotes the length of the revocation list. Parameters that are logarithmic in N, L or polynomial in the security parameter are represented as 1. Here, $0 < a < 1$ and $\epsilon > 0$ can be chosen arbitrarily. c is a large constant.

1.3 Technical Overview

We proceed to give an overview of our techniques. We begin by defining the notion of revocable predicate encryption (RPE) in both the public and secret setting, then describe the ideas used to instantiate this primitive. Finally we outline how to upgrade public/secret RPE to build trace and revoke with embedded identities with public/secret tracing.

Revocable Predicate Encryption. NWZ introduced the notion of revocable functional encryption (RFE) and used it to construct EITR with public tracing. Subsequently, Kim and Wu [40] adapted this notion to the secret key setting, under the name of revocable predicate encryption (RPE) and used it to construct EITR with secret tracing. In this work, we extend Kim and Wu's notion of RPE to the public key setting and use it to construct EITR with public tracing. Our notion of RPE in the public setting is similar to but weaker than RFE[4] – it only supports "all or nothing" decryption in contrast to RFE. This weaker notion nevertheless suffices to construct EITR and moreover admits constructions from weaker assumptions.

In RPE, the key generation algorithm takes as input the master secret key msk, a label $\mathsf{lb} \in \mathcal{L}$ and an attribute $x \in \mathcal{X}$. It outputs a secret key $\mathsf{sk}_{\mathsf{lb},x}$. The encryption algorithm takes as input the encryption key ek, a function f, a message $m \in \mathcal{M}$, and a revocation list $L \subseteq \mathcal{L}$. It outputs a ciphertext ct. Decryption recovers m if $f(x) = 1$ and $\mathsf{lb} \notin L$. In the public variant of RPE, ek is a public key, while

[4] Syntactically, RPE is "ciphertext-policy" while RFE is "key-policy", i.e. the function is emdedded in the ciphertext in RPE as against the key in RFE.

in the secret variant, ek is a secret key. In the secret variant, the scheme is also required to support a public "broadcast" functionality, i.e. there exists a public encryption algorithm that allows anyone to encrypt a message with respect to the "always-accept" policy, i.e. a policy that evaluates to true for all inputs. This is analogous to the primitive of "mixed FE" introduced by [33].

In terms of security, we require RPE to satisfy message hiding and function hiding. At a high level, message hiding stipulates that an adversary cannot distinguish between encryptions of (f, m_0) and (f, m_1) as long as every key query for (lb, x) satisfies $f(x) = 0$ or $\mathsf{lb} \in L$. Function hiding stipulates that an adversary cannot distinguish between encryptions of (f_0, m) and (f_1, m) as long as every key query for (lb, x) satisfies $f_0(x) = f_1(x)$ or $\mathsf{lb} \in L$.

Before we describe our constructions, we highlight the chief difficulties that are inherent to designing RPE:

1. *Independence of parameter sizes from $|L|$.* A key requirement in TR schemes is that the ciphertext size should be independent of the length of the revocation list L – this constraint must also be satisfied by the underlying RPE, in both the secret and public setting. In our work, we insist that even the public and secret keys satisfy $|L|$ independence. This constraint is inherited from broadcast encryption, and is challenging to satisfy. Further, note that L must be unbounded – its length cannot be fixed during setup, which introduces additional difficulties.
2. *Encrypted Computation.* While the revocation list L need not be hidden by the ciphertext, the function f[5] in the ciphertext is required to be hidden, as formalized by our function hiding requirement. Yet, this hidden function must participate in computing $f(x)$ where x is provided in the key. This requirement makes TR schemes worryingly close to collusion resistant functional encryption, an "obfustopia" primitive which we want to avoid in the secret trace setting.

Constructing Public Revocable Predicate Encryption. We proceed to describe the main ideas in constructing public RPE.

Overview of NWZ. The work of NWZ addresses the challenge of making the ciphertext size independent of $|L|$ by using a somewhere statistically binding (SSB) hash and hides the function f by using a functional encryption scheme, where f is encrypted in the ciphertext. However, they must additionally rely on iO – at a high level, this is because they require the *decryptor* to compute the SSB opening π and then run SSB verification on it (details of how SSB algorithms work are not relevant for this overview). In turn, the reason they need the decryptor to compute the opening π is because this needs both the set L and the label lb, which are available only to the decryptor – note that the encryptor has only L and the key generator has only lb. Now, since the

[5] For the informed reader, this function encodes the "index" and function hiding corresponds to "index hiding" in the literature.

decryptor has to compute π and run SSB verification, and since the program that computes SSB verification has some secrets, the decryptor is allowed to obtain obfuscation of this program. To implement this idea, they nest iO inside a compact FE scheme so that FE decryption outputs an iO which is then run by the decryptor on openings that it computes.

Trading iO *for* ABE. Above, note that the usage of iO is caused by the usage of SSB, which in turn is used to compress L. However, compression of a list has been achieved by much weaker primitives than iO in the literature of broadcast encryption – in particular, the construction of optimal broadcast encryption by Agrawal and Yamada uses the much weaker primitive of ABE (with special efficiency properties) to achieve this. However, ABE does not permit hiding anything other than a message, in particular, an ABE ciphertext cannot encrypt our function f since we desire f to participate in computation. ABE only permits computation on public values, and using ABE to encode f would force f to be public which we cannot allow.

In order to get around this difficulty, we leverage the power of functional encryption (FE), which permits encrypted computation and exactly fills the gap over ABE that we require. A natural candidate for RPE would be to simply use FE to encrypt f, L and m, and encode x and lb in the secret key for a functionality which tests that $\mathsf{lb} \notin L$, that $f(x) = 1$ and outputs m if so. Indeed, this approach using FE is folklore, and was explicitly discussed by NWZ. Yet, they end up with a construction that additionally uses SSB, iO, a puncturable PRF and secret key encryption scheme because of the requirement of size independence from $|L|$ – we do not have candidates for FE with ciphertext size independent of the public attributes. In short, ABE gives us L compactness (in some cases by encoding L in the secret key [13] and in some cases by encoding L in the ciphertext [9]) but does not hide f, whereas FE gives the opposite.

Synthesis of ABE *and* FE. We address this conundrum by combining the two primitives in a way that lets us get the best of both. In particular, we use ABE to check that $\mathsf{lb} \notin L$ and use FE to compute $f(x)$. Evidently, the two steps cannot be performed independently in order to resist mix and match attacks so we use nesting, i.e. we use FE to generate ABE ciphertexts. Here, care is required, because ABE encryption takes L as input and done naively, this strategy will again induce a size dependence on L. We address this challenge by using the special ABE by Boneh et al. [13] which enjoys succinct secret keys and encoding L in the ABE secret key. In more detail, we let the RPE encryption generate $\mathsf{ABE.sk}(C_L)$ for a circuit C_L which takes as input lb and checks that $\mathsf{lb} \notin L$. Additionally, it generates an FE ciphertext for the function f and message m. The RPE key generator computes an FE key for a function which has (lb, x) hardwired and takes as input a function f, checks whether $f(x) = 1$ and if so, generates a fresh ABE ciphertext with attribute lb and message m. Thus the decryptor can first compute FE decryption to recover the ABE ciphertext $\mathsf{ABE.ct}(\mathsf{lb}, m)$ and then use ABE decryption with $\mathsf{ABE.sk}(C_L)$ to output m if and only if $\mathsf{lb} \notin L$. It is easy to verify that this construction achieves optimal

parameters – this is because ABE has optimal parameters and we used FE only for a simple functionality that does not involve L.

Putting it all Together. The above description is over-simplified and ignores technical challenges such as how to leverage indistinguishability based security of FE, how to generate the randomness used for ABE encryption and such others – we refer the reader to Section 3 for details. However, even having filled in these details, we get only a selectively secure RPE. Substantial work and several new ideas are required for adaptive security, as we discuss next.

Adaptive Security. Next, we outline our ideas to achieve adaptive security, namely where the revocation list L is chosen adaptively by the adversary. Note that to avoid complexity leveraging, we are required to rely only on the selective security of the underling ABE – this creates multiple technical difficulties which are resolved by very carefully using specific algebraic properties of our ingredients.

Leveraging Late Generation of ABE. Our first observation is that full adaptive security of ABE may be unnecessary, since in our construction of RPE, the generation of the ABE instance is deferred until the generation of the challenge ciphertext, at which time the set of revoked users is known. This intuition turns out to be true, but via a complicated security proof as we outline next. Below, we consider the case of function hiding in the RPE ciphertext, the case of message hiding is similar.

Recall that function hiding says that two ciphertexts encoding (f_b, m, L), where $b \in \{0, 1\}$ should be indistinguishable so long as for any requested key $\mathsf{sk}_{\mathsf{lb},x}$ it holds that $f_0(x) = f_1(x)$ or $\mathsf{lb} \in L$. Note that the adversary is permitted to query for keys that allow decryption of the ciphertexts, i.e. $f_0(x) = f_1(x) = 1$.

Embedding ABE *CTs in* FE *keys.* In order to use ABE security to prove RPE security, a first (by now standard) step is to use the "trapdoor technique" [20, 7, 19], which allows us to hardwire ABE ciphertexts into FE secret keys. In the security game with the ABE challenger, the reduction submits the label lb associated with each RPE secret key as its challenge attribute and embeds the returned ABE ciphertext into the FE key. Here we immediately run into a difficulty, since in the RPE setting some ABE ciphertexts are decryptable by the adversary and we cannot leverage ABE security. Moreover, we cannot even hope to guess which keys will correspond to decryptable ABE ciphertexts since there are an unbounded polynomial number of key queries in the RPE security game. The same difficulty is faced by NWZ and is the main reason why their construction does not achieve adaptive security in the revocation list.

Polynomial Function Space Suffices for TR. To overcome this hurdle, we leverage the serendipitous fact that for the purpose of constructing TR, it suffices to construct RPE whose function space (recall that functions are encoded in the ciphertext) is only of polynomial size. This observation, which was implicitly present in [34], is abstracted and used explicitly in our proof. In particular, we

can assume that the reduction algorithm knows the challenge functions (f_0, f_1) at the beginning of the game, since it can simply guess them. Now, given the secret key query (lb, x), the reduction checks whether $f_0(x) = f_1(x)$. If yes, then there is no need to use ABE security, for the ABE ciphertexts in this case will encode the same message, and will hence be independent of the challenge bit. On the other hand, if $f_0(x) \neq f_1(x)$, then we have by the admissibility condition that $\mathsf{lb} \in L$, even when L is not known. In this case, the reduction can use the security of the ABE without any difficulty.

Additional Hurdles Stemming from ABE *Selective Security.* We now highlight another challenge in the proof. For concreteness, let us consider the second key query $(\mathsf{lb}^{(2)}, x^{(2)})$, which we assume is a pre-challenge query, and assume that $f_0(x^{(2)}) \neq f_1(x^{(2)})$. Hence, by the above discussion, we are required to use ABE security for the ciphertexts with attribute $\mathsf{lb}^{(2)}$. However, according to the selective definition, the reduction is required to choose the challenge attribute at the very start of the game, without even seeing the public parameters. At the same time, the reduction is required to simulate the ABE ciphertext for the first key query, before receiving the second key query from the adversary, that is, without seeing the ABE parameters, leading to an apparent impasse.

We address this issue by considering the following two cases separately: for the first query $(\mathsf{lb}^{(1)}, x^{(1)})$, we have (1) $f_0(x^{(1)}) \neq f_1(x^{(1)})$ or (2) $f_0(x^{(1)}) = f_1(x^{(1)})$. In first case, it is tempting to think that one can simply use a hybrid argument to change the ABE ciphertext associated with each key query satisfying $f_0(x^{(i)}) \neq f_1(x^{(i)})$ for $i \in [2]$. However, this does not work as is, since the ABE ciphertext may leak information about the ABE public key. To address this, we rely on the pseudorandomness of ciphertexts in our ABE [13] due to which we are guaranteed that the ciphertext does not reveal any information about the public parameters, enabling the hybrid strategy above. To handle the second case, we change the way in which the ABE ciphertext for the first key is generated. In more detail, we stop hardwiring the the ABE ciphertext into the first key and instead generate it directly using ABE parameters. This removes the aforementioned problem since we no longer need to embed the ABE ciphertext or public key into the first FE key. To enable this idea, we introduce additional branch of trapdoor mode for the construction to separate the paths of computation for the cases $f_0(x) = f_1(x)$ and $f_0(x) \neq f_1(x)$. To handle post-challenge queries, we need to address additional challenges, which we do not describe here. We refer the reader to Section 3 for details.

Handling Super-polynomial Revocation List. Our construction (also the secret version, described next) organically supports super-polynomially large revocation list, something that was not known before, to the best of our knowledge. In more detail, let L be a list of super-polynomial size, such that L can be represented as a string of polynomial length and there exists a circuit C_L of polynomial size which takes as input some string lb and checks whether $\mathsf{lb} \in L$ or not. Note that any super-polynomially large list must have efficient representation in order to even allow various algorithms to read it. Then, the key generation of [13] can

naturally encode the circuit C_L as before and the construction works as before. A subtlety that arises with super-polynomial L is that when we deal with post challenge key queries in the proof, we have to deal with the ABE queries in the order of key first and ciphertext later. With polynomial size L, this does not pose a problem because when the adversary chooses L, all the labels for which we use ABE security are in L and we can perform a hybrid argument over these labels. However, this is not possible for super polynomial L, which requires to rely on subexponentially secure LWE. Please see the full version of the paper [2] for details.

Instantiating Public RPE. Overall, armed with the above ideas, we get a public RPE from compact FE and efficient ABE supporting exponential sized identity space and adaptive security in the revocation list L. Currently, we only know how to instantiate our desired ABE from LWE [13], whereas FE can be instantiated in multiple different ways. A natural candidate would be the FE from standard assumptions [38, 39] which relies on pairings, LPN and low depth PRG – in this case, our RPE will require the extra assumption of LWE. Another option is to instantiate FE with a post-quantum candidate [30, 42, 49, 1, 25] from non-standard strengthenings of LWE – this has the advantage that the ABE does not incur any extra assumption in the final construction. For super-polynomial L, we need subexponential hardness of LWE in either pathway to instantiation, as discussed above.

Alternative Construction Based on Laconic OT. Here, we sketch an alternative construction of RPE based on laconic OT (LOT) [22] that works when the number N of possible labels is polynomially bounded (i.e., the identity space is of polynomial size). Since LOT is known to be possible from various assumptions, this diversifies the assumptions that we need to rely on. The basic idea is to replace ABE with LOT. In more detail, the encryptor chooses LOT parameters instead of ABE parameters and computes the digest of the list of recipients (or equivalently, the list of revoked users), which is represented as a binary string of length N with 1 for non-revoked identities. The digest, whose size is independent of N, is then embedded into the FE ciphertext. Then, FE decryption yields LOT encryption of the message for the label $\mathsf{lb} \in [N]$, which is the label associated with the secret key, instead of ABE ciphertext. The LOT ciphertext is encrypted so that it can be decrypted only when the lb-th bit of the binary string representing the list of recipients is 1. We note that this idea does not extend for identities from exponentially large space and cannot therefore support embedded identities any more.

Revocable Predicate Encryption in Private Setting. For private revocable predicate encryption, our starting point is the work of Goyal et al. [35], who show how to combine "broadcast mixed FE" (called BMFE) together with ABE to achieve RPE (via a different abstraction which they call AugBE). They construct BMFE by adding the broadcast functionality to the primitive of mixed FE defined by [33]. They embed BMFE ciphertext into an ABE ciphertext to achieve RPE, where BMFE is constructed from LWE and ABE is instantiated using pairings.

Supporting Exponential Identity Space. To begin, we upgrade their notion of BMFE to support an exponential space of identities (which we refer to as labels) towards the goal of embedded identity trace and revoke. We refer to our notion as Revocable Mixed FE (denoted by RMFE) and construct it from LWE. Both [35] and our work start with a mixed FE scheme and add broadcast to it, but their construction builds upon the scheme based on constrained PRFs [21] while ours begins with the scheme based on Lockable Obfuscation (LO), also from [21]. Our construction of RMFE deviates significantly from theirs, and achieves significantly better secret key size – $O(\log N)$ as against $O(N)$ – in addition to supporting exponential instead of polynomial space. We describe this construction next.

Mixed FE. The notion of mixed FE was introduced by Goyal, Koppula and Waters in the context of traitor tracing [33]. Identifying and constructing this clever primitive is the key insight that enables [33] to construct traitor tracing with optimal parameters from LWE. Mixed FE is, as the name suggests, a mix of public and secret key FE. Thus, it has a secret as well as a public encryption procedure. The secret encryption procedure takes as input a function f and computes ct_f. This is decryptable by a key sk_x to recover $f(x)$. The adversary can make one query to the encryption oracle in addition to getting the challenge ciphertext for challenge (f_0, f_1). It can also make an unbounded number of key requests so long as $f_0(x) = f_1(x)$. The public encryption algorithm computes a ciphertext for the "always accept" function, i.e. a function which evaluates to 1 for any input x. It is required that the public ciphertext be indistinguishable from the secret ciphertext.

One of the constructions of mixed FE suggested by [21] uses a secret key FE scheme (SKFE) to construct the secret encryption algorithm and leverages the power of lockable obfuscation (LO) to construct the public encryption procedure. Recall that in a lockable obfuscation scheme [32, 50] there exists an obfuscation algorithm Obf that takes as input a program C, a message m and a (random) "lock value" α and outputs an obfuscated program $\tilde{P}$. One can evaluate the obfuscated program on any input x to obtain as output m if $P(x) = \alpha$ and $\perp$ otherwise. Intuitively, the idea of [21] is to wrap the FE ciphertext using LO and to define the public key encryption algorithm as outputting a simulated version of the LO obfuscated circuit, which is publicly sampleable.

In more detail, the construction works as follows. The secret key for a user with input x is an SKFE secret key $\mathsf{SKFE.sk}(x)$. The secret ciphertext of MFE for function f is constructed as follows.

1. First, SKFE ciphertext $\mathsf{SKFE.ct}(H_{f,\alpha})$ is generated, where α is a freshly chosen random value and $H_{f,\alpha}$ is a circuit that takes as input x and outputs α if $f(x) = 0$ and 0 otherwise.
2. Then, LO with lock value α and any message $m \neq \perp$ is used to obfuscate the circuit $\mathsf{SKFE.Dec}(\mathsf{SKFE.ct}(H_{f,\alpha}), \cdot)$, namely the circuit that takes as input an SKFE secret key and decrypts the hardwired ciphertext using this.

The decryption result of MFE is defined as 1 if the evaluation result of the LO circuit on the given input SKFE secret key is $\perp$ and 0 otherwise. Correctness follows from correctness of SKFE and LO. In particular, if $f(x) = 0$, then SKFE decryption outputs α, which unlocks the LO to give m, otherwise $\perp$. By definition, MFE decryption will output 1 if LO outputs $\perp$ which happens when $f(x) = 1$, and 0 otherwise.

Revocable Mixed FE. RMFE augments MFE so that the encryption algorithms (both secret and public) now include a revocation list L and the secret key additionally includes a label lb. A secret key $\mathsf{sk}_{\mathsf{lb},x}$ decrypts a secret ciphertext $\mathsf{ct}_{f,L}$ to recover $f(x)$ if $\mathsf{lb} \notin L$ and 1 otherwise. For a public ciphertext ct_L, the output of decryption is always 1 regardless of which secret key is being used. For security, we need two properties: function hiding and mode hiding. For function hiding, we require that a secret ciphertext $\mathsf{ct}_{f_0,L}$ is indistinguishable from $\mathsf{ct}_{f_1,L}$ if for all queries, either $f_0(x) = f_1(x)$ or $\mathsf{lb} \in L$. For mode hiding, we require that a secret ciphertext $\mathsf{ct}_{f,L}$ is indistinguishable from a public ciphertext ct_L. Recall that L is not required to be hidden, but we require that the parameters do not depend on $|L|$.

To extend MFE to RMFE, we retain the idea of letting the secret ciphertext be an LO obfuscated circuit and public ciphertext be the simulated LO. To incorporate the list L, we must ensure that the LO lock value α is recovered only when $f(x) = 0$ and $\mathsf{lb} \notin L$. To do so, we consider two subsystems such that one system outputs partial decryption result α_1 only when $f(x) = 0$ and the second system outputs partial decryption result α_2 only when $\mathsf{lb} \notin L$ such that $\alpha = \alpha_1 + \alpha_2$. We must ensure that α_1 and α_2 are user specific decryption results to avoid collusion attacks.

Note that the second subsystem, which entails L, should be constructed so that the hardwired values inside the circuit do not depend on $|L|$, but still control access to the value α_2 depending on L. To satisfy these apparently conflicting requirements, we make use of the unique algebraic properties of the ABE construction by Boneh et al [13], as described below. For the first subsystem, we use SKFE.

In more detail, our candidate scheme is as follows.

1. **Secret Key:** The RMFE secret key consists of $\mathsf{ABE.ct}(\mathsf{lb}, K)$ and $\mathsf{SKFE.ct}((x, K, R))$ where K and R are user specific random strings, lb is used as an attribute and K is the plaintext for ABE encryption.
2. **Ciphertext:** To generate RMFE ciphertext, the secret key encryption procedure is as follows:
   - It first generates $\mathsf{ABE.sk}(C_L)$, where C_L is a circuit that takes as input a label lb and outputs 1 only when $\mathsf{lb} \notin L$.
   - It also generates $\mathsf{SKFE.sk}(H_{f,\alpha})$, where $H_{f,\alpha}$ takes as input (x, K, R) and outputs $K \oplus \alpha$ if $f(x) = 0$ and R if $f(x) = 1$.
   - Now, consider the circuit $CC[\mathsf{ABE.sk}(C_L), \mathsf{SKFE.sk}(H_{f,\alpha})]$, which takes as an input the pair (ABE.ct, SKFE.ct), decrypts both ABE and SKFE ciphertexts using their respective keys, and then outputs the XOR between the decryption results.

– The final ciphertext is an LO of $CC[\mathsf{ABE.sk}(C_L), \mathsf{SKFE.Enc}(H_{f,K,\alpha})]$ with lock value α and any arbitrary message $m \neq \perp$.

By key compactness of [13], the size of $\mathsf{ABE.sk}(C_L)$ is independent of $|L|$. A subtle point here is that ABE decryption is happening inside the LO and this depends on L. If the LO must process L, then the size of the LO and hence ciphertext blows up with L! Fortunately, the algebraic structure of the ABE scheme we use [13] again comes to our rescue. At a high level, ABE decryption can be divided into an "L-dependent" step which results in a short processed ciphertext, followed by an "L-independent" step. Importantly, the L-dependent step does not depend on the ABE secret key which is hardwired in the LO and hence inaccessible, and can hence be performed *outside* the LO by the decryptor! The resultant short processed ciphertext can then be provided as input to the LO preventing the problematic size blowup.

RMFE *Proof Overview.* Next we outline some of the ideas developed for the security proof. For ease of understanding, we limit ourselves to the simpler setting where the adversary does not have access to the encryption oracle. This restriction can be removed using combinatorial tricks, similar to [21]. For security, we must argue two properties – mode indistinguishability and function hiding. The former can be established by relying on security of SKFE and LO analogously to the MFE proof in [21]. Hence, we focus on function hiding for the rest of the overview, which is subtle and requires several new ideas.

For function hiding, we must make use of the security of ABE and SKFE. Intuitively, security of SKFE guarantees that the values encoded in SKFE ciphertexts and secret keys are hidden, beyond what is revealed by decryption.[6] First note that given a key for (lb, x) such that $f_0(x) = f_1(x)$, no information about the challenge bit is revealed by decryption, since the decryption results of SKFE are the same for both cases. The case with $f_0(x) \neq f_1(x)$ is more challenging. Let us assume $f_0(x) = 0$ and $f_1(x) = 1$. In this case, the decryption result of the challenge ciphertext is R or $K \oplus \alpha$ depending on the value of the challenge bit. Since both are random strings, it is tempting to conclude that they do not reveal any information of the challenge bit.

However, in reality, information about K is encoded in the ciphertext $\mathsf{ABE.ct}(\mathsf{lb}, K)$ and creates a correlation which must be handled. Indeed, a computationally unbounded attacker can learn the challenge bit by breaking open the ABE ciphertext, recovering K and then correlating it with the decryption result of SKFE. Hence, security of ABE must play a role and fortunately, we show that security of ABE suffices to overcome this difficulty. Recall that our security definition of RMFE requires that if $f_0(x) \neq f_1(x)$, then it should hold that $\mathsf{lb} \in L$. This means that the ciphertext $\mathsf{ABE.ct}(\mathsf{lb}, K)$ is computationally indistinguishable from $\mathsf{ABE.ct}(\mathsf{lb}, 0)$, since the only ABE secret key available to the adversary is $\mathsf{ABE.sk}(C_L)$. Now, in the adversary's view, both $K \oplus \alpha$ and R are random strings that are independent from other parameters.

[6] We note that we need message *and* function hiding security for the underlying SKFE, while [21] only needs message hiding security.

Therefore, the adversary cannot obtain any information of the challenge bit from the decryption result in this case as well. For more details, please see Section 4.

Comparison with the BMFE *by Goyal et al. [35].* We observe that both our RMFE as well as the BMFE by [35] rely solely on LWE. However, our secret key is $\mathsf{ABE.ct}(\mathsf{lb}, K)$ and $\mathsf{SKFE.ct}((x, K, R))$, which has optimal size, being clearly independent of N and L. In contrast their secret key depends linearly on N. We also observe that our RMFE can support an exponentially large space of identities, while their BMFE does not.

Combining RMFE *and* ABE *to get* RPE. Finally, we nest our RMFE inside an outer ABE scheme to obtain RPE. This step is very similar to [35], but we need to use a different ABE scheme. In particular, in the construction of RPE in [35], a key policy ABE (kpABE) is used to encrypt the message m with attributes as the RMFE ciphertext along with the list L. The RPE secret key for (x, lb) is a kpABE secret key for a the RMFE decryption circuit $\mathsf{RMFE.Dec}(\mathsf{RMFE.sk}, \cdot, \cdot)$.

An obvious difficulty here is that encoding the attribute $(L, \mathsf{RMFE.ct})$ in the ABE ciphertext can cause the ciphertext size to depend on the size of L. To avoid this blowup, [35] use a special kpABE which has the property that the ciphertext size is independent of the size of the attribute. They instantiate this kpABE with the scheme [9] which uses pairings[7]. However, we cannot use [9, 45] because of the following two reasons:

1. First, the ABE scheme by [9] only supports NC_1. However, our circuit $\mathsf{RMFE.Dec}(\mathsf{RMFE.sk}, \cdot, \cdot)$ does not fit into NC_1[8].
2. Furthermore, even if the above problem could be resolved, using [9] is problematic since their ABE has secret and public keys at least as large as $O(|L|)$. While the scheme of [35] also suffers from this blow-up, our goal is to obtain short keys, independent of $|L|$.

The first problem cannot be resolved even if we use the ABE schemes for circuits [31, 13], since their ciphertext size also depends on $|L|$. To instantiate our ABE, we use recent construction of compact cpABE from evasive and tensor LWE [48], whose parameter sizes depend only on the input length of the circuit and are independent of its size. Armed with the above ideas, we suggest the following RPE:

1. The encryption algorithm of RPE, given m, f, L computes RMFE ciphertext encoding (f, L) and then computes $\mathsf{cpABE.Enc}(\mathsf{RMFE.Dec}(\mathsf{RMFE.ct}, \cdot, L), m)$.
2. The key generation algorithm RPE given (lb, x), computes RMFE secret key for (lb, x) and outputs $\mathsf{cpABE.sk}(\mathsf{RMFE.sk})$.

[7] In fact, one could instead use the kpABE constructed by [45]. This enjoys the same efficiency properties and is based on the standard DLIN assumption as against the q-type assumption of [9].

[8] The informed reader may wonder whether we can solve this issue by using preprocessing as in [35] but this does not work due to technical reasons.

Correctness of RPE follows from correctness of cpABE and RMFE while optimality of parameters follows from the efficiency of the underlying schemes. In particular, observe that all parameters are independent of $|L|$. Also note that evasive and tensor LWE are required only to instantiate cpABE with the desired efficiency. If future work standardizes the assumptions underlying the cpABE, our construction will inherit these assumptions. For more details, we refer the reader to the full version of the paper [2, Sec. 6].

Instantiating Secret RPE. Currently, the only two suitable ABE schemes that we know to instantiate our compiler are the LWE based kpABE by Boneh et al. [13] and the evasive and tensor LWE based cpABE by Wee [48]. These two ABEs give us a secret RPE scheme supporting exponential identities and with optimal parameters, from evasive and tensor LWE. Note that this construction does *not* achieve adaptive security in the revoke list. Nevertheless, it is the first construction of optimal RPE, even without embedded identities, from any assumption outside Obfustopia. Note that the usage of a non-standard assumption outside of obfustopia (in particular, only from lattice techniques) is somewhat inherent given that even broadcast encryption *without* tracing requires non-standard assumption if we instantiate it only from lattices. We are hopeful that future improvements in cpABE will yield a construction from completely standard assumptions.

Trace and Revoke with Optimal Ciphertext from LWE and Pairings. Along the way, we observe that the broadcast and trace construction provided by Goyal et al. [35], without embedded identities, can be easily modified to achieve at least optimal ciphertext size, from the same assumptions. At a high level, they construct a broadcast mixed FE from LWE with optimal ciphertext size and then nest this inside the kpABE by [9], which enjoys ciphertext size independent of the attribute length, and can support computation in NC_1. Since their BMFE decryption does not fit into NC_1, they preprocess the ciphertext so that part of the decryption is performed "outside", namely, they group $\log N$ matrix tuples into c groups of $(\log N)/c$ tuples each. Then they precompute all possible $2^{(\log N)/c} = N^{1/c}$ subset-products within each group. Due to this, BMFE decryption only needs to multiply together c of the preprocessed matrices, which can be done in NC_1 so long as c is constant. Unfortunately, this step increases their ciphertext size to $O(N^\epsilon)$ for any $\epsilon > 0$ though the BMFE ciphertext size was optimal.

We observe that they are "under-using" the ciphertext size independence of [9] – in particular, while the attribute length has indeed been blown up to $O(N^\epsilon)$, this does not affect the ciphertext size of [9]. Moreover, while the attribute must also be provided outside in the clear, this part can be compressed, i.e. the preprocessing which expands the attribute to size N^ϵ can be performed by the decryptor directly by grouping and multiplying matrices as described above, and there is no need for the encryptor to provide this expanded form. Thus, their scheme tweaked with this simple modification already achieves ciphertext of optimal size, though with large secret key $O(N^c)$ for some large constant c.

Trace and Revoke from Revocable Predicate Encryption. It remains to show how to construct the final goal of trace and revoke with embedded identities. As discussed earlier, we follow [44, 40] and use the abstraction of RPE to build trace and revoke. However, to embed identities in our trace and revoke schemes, we deviate from these works and instead build upon ideas developed by [34] (henceforth GKW) in the context of traitor tracing.

Embedded Identity Traitor Tracing (EITT) by GKW. The work of Goyal, Koppula and Waters [34] provided an alternative approach for embedding identities in traitor tracing schemes. A well known approach for constructing Traitor Tracing systems suggested by Boneh, Sahai and Waters [11] is via the intermediate primitive of *Private Linear Broadcast Encryption* (PLBE), which allows to construct a tracing algorithm that performs a linear search over the space of users to recover the traitor. Since the number of users was polynomial, this algorithm could be efficient. However, if we allow arbitrary identities to correspond to user indices then the space over which this search must be performed becomes exponential even if the number of users is polynomial, and the trace algorithm is no longer efficient. The main new idea in NWZ that enables them to handle exponentially large identity spaces is to replace a linear search over indices by a clever generalization of binary search, which efficiently solves an "oracle jump problem" which in turn suffices for tracing.

Goyal, Koppula and Waters (GKW) provided an alternate route to the problem of embedding identities. Instead of using PLBE and generalizing the search procedure, they instead extend the definition of PLBE to support embedded identities, denoted by EI-PLBE, and then used this to get a full fledged EITT scheme. This approach has the notable advantage that even if the space of identities is exponential, it can use the fact that the number of users is only *polynomial* and hence rely on only *selective* security of the underlying primitives. In particular, they demonstrate a "nested" tracing approach, where the tracing algorithm works in two steps: first, it outputs a set of indices that correspond to the users that are traitors, and then it uses each index within this set to recover the corresponding identity. Additionally, GKW provide a sequence of (increasingly stronger) TT primitives with embedded identities, namely, indexed EITT, bounded EITT and finally unbounded EITT where unbounded EITT satisfies the most general notion of embedded identity traitor tracing. They also provide generic transformations between these notions, which allows to focus on the weakest notion for any new instantiation.

Embedded Identity Trace and Revoke (EITR). We adapt the approach of GKW and show how to use their nested approach to trace embedded identities even in the more challenging setting of trace and revoke. As in their case, this lets us use polynomial hardness assumptions in obtaining EITR, in contrast to NWZ. We also define indexed, bounded and unbounded EITR and provide transformations between them. Our definitions as well as transformations are analogous to GKW albeit care is required to incorporate the revoke list L in each step and adapt the

definitions and proofs of security accordingly. We then construct indexed EITR using RPE, and obtain unbounded EITR via our generic conversions.

We note that our framework unifies the approaches of Kim and Wu [40] who used the framework of RPE in the context of TR and that of GKW who used the framework of EI-PLBE in the context of TT, to obtain EITR. This unification yields a clean abstraction which can be used for both public and secret key settings. We believe this framework is of independent interest. We refer the reader to Sections 7, 8, 9 and 10 in the full version [2] for details. An overview of our constructions is provided in Figure 1.

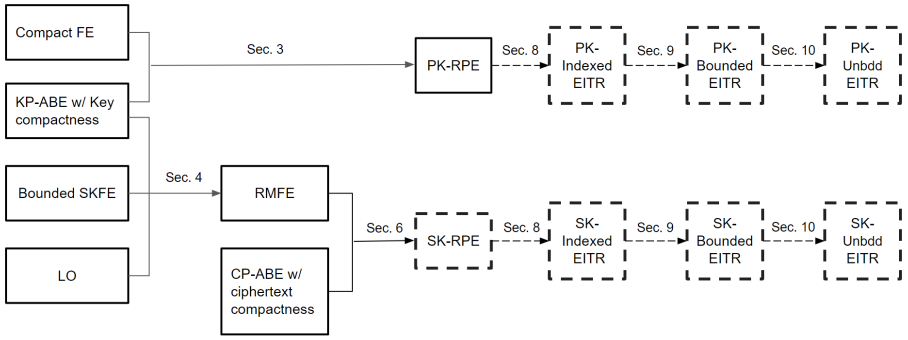

Fig. 1. Overview of our constructions. Solid lines represent the implications shown by our work and are based on new techniques. Dashed lines represent the implications that are new but based on techniques developed in [34]. The constructions in dashed boxes are provided in the respective sections of the full version of our paper [2].

Organization of the paper. We define RPE in Section 2 and construct public-key RPE in Section 3. We provide our construction of RMFE in Section 4. Due to space constraints, our construction of secret-key RPE using RMFE is deferred to the full version [2]. We also refer the reader to the full version [2] for preliminaries, definitions of indexed-EITR, bounded-EITR and unbounded-EITR and conversions between them [2, Sec. 8, Sec. 9, Sec. 10], and the mechanism for supporting super-polynomial sized revocation list [2, Sec. 11].

2 Revocable Predicate Encryption

We define revocable predicate encryption (RPE), in both public and secret key setting. Since the two notions differ only in the encryption algorithm, we present them here in a unified way.

Definition 2.1. A RPE scheme for an attribute space $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in[\mathbb{N}]}$, a function family $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in[\mathbb{N}]}$ where $\mathcal{F}_\lambda = \{f : \mathcal{X}_\lambda \to \{0,1\}\}$, a label space $\mathcal{L} =$

$\{\mathcal{L}_\lambda\}_{\lambda\in[\mathbb{N}]}$ and a message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in[\mathbb{N}]}$ has the following probabilistic polynomial time algorithms:

- $\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$. The setup algorithm takes the security parameter λ as input and it outputs a master public key mpk and a master secret key msk.
- $\mathsf{KeyGen}(\mathsf{msk}, \mathsf{lb}, x) \to \mathsf{sk}_{\mathsf{lb},x}$[9]. The key generation algorithm takes as input the master secret key msk, a label $\mathsf{lb} \in \mathcal{L}_\lambda$ and an attribute $x \in \mathcal{X}_\lambda$. It outputs a secret key $\mathsf{sk}_{\mathsf{lb},x}$.
- $\mathsf{Enc}(\mathsf{ek}, f, m, L) \to \mathsf{ct}$. The encryption algorithm takes as input the encryption key ek, a function f, a message $m \in \mathcal{M}_\lambda$, and a revocation list $L \subseteq \mathcal{L}_\lambda$. It outputs a ciphertext ct.
- $\mathsf{Dec}(\mathsf{sk}_{\mathsf{lb},x}, \mathsf{ct}, L) \to m'$. The decryption algorithm takes the secret key $\mathsf{sk}_{\mathsf{lb},x}$, a ciphertext ct, and a revocation list L and it outputs $m' \in \mathcal{M}_\lambda \cup \{\perp\}$.

In *public-key RPE*, we take $\mathsf{ek} = \mathsf{mpk}$ in the Enc algorithm, and in *secret-key RPE*, we take $\mathsf{ek} = \mathsf{msk}$. Furthermore, there is an additional algorithm in the secret key setting defined as follows:

- $\mathsf{Broadcast}(\mathsf{mpk}, m, L) \to \mathsf{ct}$. On input the master public key, a message m, and a revocation list $L \subseteq \mathcal{L}_\lambda$, the broadcast algorithm outputs a ciphertext ct.

Definition 2.2 (Correctness). *A revocable predicate encryption scheme is said to be correct if there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*, label* $\mathsf{lb} \in \mathcal{L}_\lambda$*, attributes* $x \in \mathcal{X}_\lambda$*, predicates* $f \in \mathcal{F}_\lambda$ *such that* $f(x) = 1$*, all messages* $m \in \mathcal{M}_\lambda$ *and any set of revoked users* $L \subseteq \mathcal{L}_\lambda$ *such that* $\mathsf{lb} \notin L$*, if we set* $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and* $\mathsf{sk}_{\mathsf{lb},x} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{lb}, x)$*, then the following holds*

$$\mathsf{Pr}\left[\, \mathsf{Dec}(\mathsf{sk}_{\mathsf{lb},x}, \mathsf{ct}, L) = m \,\right] \geq 1 - \mathsf{negl}(\lambda),$$

for $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{ek}, f, m, L)$ *(Encryption correctness) and* $\mathsf{ct} \leftarrow \mathsf{Broadcast}(\mathsf{mpk}, m, L)$ *(Broadcast correctness).*

Security. In the following security definitions, we assume for simplicity that the adversary does not make key queries for same input (lb, x) more than once.

Definition 2.3 (q-query Message Hiding). *Let* $q(\cdot)$ *be any fixed polynomial. A* RPE *scheme satisfies q-query message hiding property if for every PPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for every* $\lambda \in \mathbb{N}$*, all messages* $m \in \mathcal{M}_\lambda$ *and any subset of revoked users* $L \subseteq \mathcal{L}_\lambda$*, the following holds*

$$\mathsf{Pr}\left[\begin{array}{l} \beta' = \beta : \\ \end{array} \begin{array}{l} (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda); \\ (f, m_0, m_1, L) \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot),\mathsf{Enc}(\mathsf{ek},\cdot,\cdot,\cdot)}(\mathsf{mpk}); \\ \beta \leftarrow \{0,1\}; \mathsf{ct}_\beta \leftarrow \mathsf{Enc}(\mathsf{ek}, f, m_\beta, L); \\ \beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot),\mathsf{Enc}(\mathsf{ek},\cdot,\cdot,\cdot)}(\mathsf{ct}_\beta) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathcal{A}$ *can make at most* $q(\lambda)$ *queries to the encryption oracle* $\mathsf{Enc}(\mathsf{ek}, \cdot, \cdot, \cdot)$*, and* $\mathcal{A}$ *is admissible if and only if for all the key queries* (lb, x) *to the* $\mathsf{KeyGen}(\mathsf{msk}, \cdot, \cdot)$ *oracle, either* $f(x) = 0$ *or* $\mathsf{lb} \in L$*.*

[9] We want to point out that the secret key $\mathsf{sk}_{\mathsf{lb},x}$ does not hide the corresponding label lb and attribute x and we assume these to be included in the secret key.

Definition 2.4 (q-query Selective Message Hiding). *This is the same as the Def 2.3 except that $\mathcal{A}$ outputs the revocation list L in the beginning of the game, before the* Setup *algorithm is run.*

Definition 2.5 (q-query Very Selective Message Hiding). *This is the same as the Def 2.4 except that $\mathcal{A}$ outputs all the key queries (Ib, x) to the $\mathsf{KeyGen}(\mathsf{msk}, \cdot, \cdot)$ oracle in the beginning of the game, before the* Setup *algorithm is run.*

Definition 2.6 (q-query Function Hiding). *Let $q(\cdot)$ be any fixed polynomial. A* RPE *scheme satisfies q-query function hiding property if for every PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\lambda \in \mathbb{N}$, all messages $m \in \mathcal{M}_\lambda$ and any subset of revoked users $L \subseteq \mathcal{L}_\lambda$, the following holds*

$$\mathsf{Pr}\left[\beta' = \beta : \begin{array}{l}(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda);\\ (f_0, f_1, m, L) \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot),\mathsf{Enc}(\mathsf{ek},\cdot,\cdot,\cdot)}(\mathsf{mpk});\\ \beta \leftarrow \{0,1\}; \mathsf{ct}_\beta \leftarrow \mathsf{Enc}(\mathsf{ek}, f_\beta, m, L);\\ \beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot),\mathsf{Enc}(\mathsf{ek},\cdot,\cdot,\cdot)}(\mathsf{ct}_\beta)\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathcal{A}$ can make at most $q(\lambda)$ queries to the encryption oracle $\mathsf{Enc}(\mathsf{ek}, \cdot, \cdot, \cdot)$, and $\mathcal{A}$ is admissible if and only if for all the key queries (Ib, x) to the $\mathsf{KeyGen}(\mathsf{msk}, \cdot, \cdot)$ oracle, either $f_0(x) = f_1(x)$ or $\mathsf{Ib} \in L$.

Definition 2.7 (q-query Selective Function Hiding). *This is the same as the Def 2.6 except that $\mathcal{A}$ outputs the revocation list L in the beginning of the game, before the* Setup *algorithm is run.*

The following security notion is defined only for secret-key RPE scheme.

Definition 2.8 (q-query Selective Broadcast Security). *Let $q(\cdot)$ be any fixed polynomial. A* RPE *scheme satisfies q-query selective broadcast security if there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every PPT adversary $\mathcal{A}$, for every $\lambda \in \mathbb{N}$, all messages $m \in \mathcal{M}_\lambda$ and any subset of revoked users $L \subseteq \mathcal{L}_\lambda$, the following holds*

$$\mathsf{Pr}\left[\beta' = \beta : \begin{array}{l}L \leftarrow \mathcal{A}(1^\lambda);\\ (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda);\\ f, m \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot),\mathsf{Enc}(\mathsf{msk},\cdot,\cdot,\cdot)}(\mathsf{mpk});\\ \beta \leftarrow \{0,1\}; \mathsf{ct}_0 \leftarrow \mathsf{Enc}(\mathsf{msk}, f, m, L);\\ \mathsf{ct}_1 \leftarrow \mathsf{Broadcast}(\mathsf{mpk}, m, L);\\ \beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot),\mathsf{Enc}(\mathsf{msk},\cdot,\cdot,\cdot)}(\mathsf{ct}_\beta)\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathcal{A}$ can make at most $q(\lambda)$ queries to the encryption $\mathsf{Enc}(\mathsf{msk}, \cdot, \cdot, \cdot)$ oracle and $\mathcal{A}$ is admissible if and only if $f(x) = 1, \forall x \in \mathcal{X}_\lambda$.

Remark 2.9. In the *public-key* RPE scheme, the adversary $\mathcal{A}$ can itself simulate the encryption oracle $\mathsf{Enc}(\mathsf{ek}, \cdot, \cdot, \cdot)$, as $\mathsf{ek} = \mathsf{mpk}$ in this setting. Therefore, in public-key setting, we refer to the security definitions without imposing the q-query bound on the encryption oracle.

Remark 2.10. We note that when the message space is binary, function space $\mathcal{F}_\lambda$ is polynomially small and q is a constant, the weaker security definitions where adversary outputs the challenge function f, the challenge message m and the SK-Enc query functions $\{\bar{f}_i\}_{i\in[q]}$ at the beginning of the game, before the $\mathsf{Setup}(1^\lambda)$ algorithm is run, is equivalent to the definitions where the adversary outputs $f, m, \{\bar{f}_i\}_{i\in[q]}$ adaptively. First, the functions can be guessed with polynomial loss. Furthermore, if we restrict the message space to be binary, we can guess the challenge message as well. To extend the message space, we can encrypt each bit by parallel systems.

3 Public-key RPE from FE and LWE

In this section we provide our construction of a public key RPE scheme $\mathsf{RPE} = (\mathsf{RPE.Setup}, \mathsf{RPE.KeyGen}, \mathsf{RPE.Enc}, \mathsf{RPE.Dec})$ for an attribute space $\mathcal{X} = \{\mathcal{X}_\lambda\}_\lambda$, a function family $\mathcal{F} = \{\mathcal{F}_\lambda\}_\lambda$ where $\mathcal{F}_\lambda = \{f : \mathcal{X}_\lambda \to \{0,1\}\}$, a label space $\mathcal{L} = \{\mathcal{L}_\lambda\}_\lambda$ and a message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_\lambda$ from polynomial hardness assumptions. We assume that $|\mathcal{F}_\lambda|$ and $|\mathcal{M}_\lambda|$ are bounded by some polynomial in λ. The restriction on $|\mathcal{F}_\lambda|$ is sufficient for our purpose and the restriction on $|\mathcal{M}_\lambda|$ can be removed by running the scheme in parallel.
Our construction uses the following building blocks:

1. A Sel-INDr secure key-policy ABE scheme $\mathsf{kpABE} = (\mathsf{kpABE.Setup}, \mathsf{kpABE.Enc}, \mathsf{kpABE.KeyGen}, \mathsf{kpABE.Dec})$ for circuit class $\mathcal{C}_{\ell(\lambda),d(\lambda)}$ with parameter succinctness and key compactness ([2, Theorem 2.18]). Here $\ell(\lambda)$ is the input length and is the length of labels in our setting and the depth of the circuit is $d(\lambda) \in \omega(\log \lambda)$ to support unbounded revocation list. The message space of the scheme kpABE is $\mathcal{M} = \{\mathcal{M}_\lambda\}_\lambda$ and $\mathcal{CT}_{\mathsf{kpABE}}$ denotes the ciphertext space. We assume that uniform sampling from $\mathcal{CT}_{\mathsf{kpABE}}$ is efficiently possible without any parameter.
2. A (fully) compact, selectively secure, public-key functional encryption scheme $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.Enc}, \mathsf{FE.KeyGen}, \mathsf{FE.Dec})$ that supports polynomial sized circuits. We assume that the message space is sufficiently large so that it can encrypt an ABE master public key, a (description of) function $f \in \mathcal{F}_\lambda$, a PRF key, two secret keys of SKE, and a trit $\mathsf{mode} \in \{0,1,2\}$.
3. A PRF $F : \{0,1\}^\lambda \times \mathcal{X} \to \{0,1\}^t$ where t is the length of the randomness used in kpABE encryption ([2, Def. 2.1]).
4. A symmetric key encryption schemes $\mathsf{SKE} = (\mathsf{SKE.KeyGen}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ with pseudorandom ciphertexts ([2, Def. 2.3]).We let $\mathcal{CT}_{\mathsf{SKE}}$ denote the ciphertext space of SKE.[10] We assume that uniform sampling from $\mathcal{CT}_{\mathsf{SKE}}$ is efficiently possible without any parameter.

We describe our construction below.

[10] We note that we use the same ciphertext space for simplicity even though messages with different lengths are going to be encrypted. To have the same ciphertext space, we can, for example, pad short messages to be some fixed length, which is possible when the message length is bounded by some polynomial.

$\mathsf{RPE.Setup}(1^\lambda) \to (\mathsf{RPE.mpk}, \mathsf{RPE.msk})$. The setup algorithm does the following:
- Generate $(\mathsf{FE.mpk}, \mathsf{FE.msk}) \leftarrow \mathsf{FE.Setup}(1^\lambda)$.
- Output $\mathsf{RPE.mpk} = \mathsf{FE.mpk}$ and $\mathsf{RPE.msk} = \mathsf{FE.msk}$.

$\mathsf{RPE.KeyGen}(\mathsf{RPE.msk}, \mathsf{lb}, x) \to \mathsf{RPE.sk}_{\mathsf{lb},x}$. The key generation algorithm does the following:
- Sample random values $\gamma_1, \gamma_2, \delta \leftarrow \mathcal{CT}_{\mathsf{SKE}}$.
- Construct a circuit $\mathsf{Re\text{-}Enc}[\mathsf{lb}, x, \gamma_1, \gamma_2, \delta]$ which has the label lb, attribute x, γ_1, γ_2 and δ hardwired, as defined in Figure 2.
- Compute $\mathsf{FE.sk}_{\mathsf{lb},x} \leftarrow \mathsf{FE.KeyGen}(\mathsf{FE.msk}, \mathsf{Re\text{-}Enc}[\mathsf{lb}, x, \gamma_1, \gamma_2, \delta])$.
- Output $\mathsf{RPE.sk}_{\mathsf{lb},x} = \mathsf{FE.sk}_{\mathsf{lb},x}$.

$\mathsf{RPE.Enc}(\mathsf{RPE.mpk}, f, m, L) \to \mathsf{RPE.ct}$. The encryption algorithm does the following:
- Parse $\mathsf{RPE.mpk} = \mathsf{FE.mpk}$.
- Sample a PRF key $K \leftarrow \{0,1\}^\lambda$.
- Generate $(\mathsf{kpABE.mpk}, \mathsf{kpABE.msk}) \leftarrow \mathsf{kpABE.Setup}(1^\lambda)$.
- Compute $\mathsf{FE.ct} \leftarrow \mathsf{FE.Enc}(\mathsf{FE.mpk}, (\mathsf{kpABE.mpk}, f, m, K, 0, \perp, \perp))$.
- Construct a circuit C_L, with revocation list L hardwired defined as follows:
  On input a label $\mathsf{lb} \in \mathcal{L}_\lambda$,

$$C_L(\mathsf{lb}) = 1 \text{ if and only if } \mathsf{lb} \notin L. \tag{3.1}$$

  Compute $\mathsf{kpABE.sk}_L \leftarrow \mathsf{kpABE.KeyGen}(\mathsf{kpABE.msk}, C_L)$.
- Output $\mathsf{RPE.ct} = (\mathsf{kpABE.mpk}, \mathsf{kpABE.sk}_L, \mathsf{FE.ct})$.

$\mathsf{RPE.Dec}(\mathsf{RPE.sk}_{\mathsf{lb},x}, \mathsf{RPE.ct}, L) \to m'$. The decryption algorithm does the following:
- Parse $\mathsf{RPE.ct} = (\mathsf{kpABE.mpk}, \mathsf{kpABE.sk}_L, \mathsf{FE.ct})$ and $\mathsf{RPE.sk}_{\mathsf{lb},x} = \mathsf{FE.sk}_{\mathsf{lb},x}$.
- Compute $\mathsf{ct}' = \mathsf{FE.Dec}(\mathsf{FE.sk}_{\mathsf{lb},x}, \mathsf{FE.ct})$.
- Construct circuit C_L from L and compute $m' = \mathsf{kpABE.Dec}(\mathsf{kpABE.mpk}, \mathsf{kpABE.sk}_L, C_L, \mathsf{ct}', \mathsf{lb})$.
- Output m'.

Correctness and Security. We prove that our construction of RPE satisfies correctness and both function hiding and message hiding security via the following theorems.

Theorem 3.1. *Suppose* FE *and* kpABE *schemes are correct. Then the above construction satisfies the encryption correctness (Def. 2.2).*

Theorem 3.2. *Assume that F is a secure* PRF, SKE *is correct and secure,* FE *and* kpABE *are secure as per Definition [2, Def. 2.6] and [2, Def. 2.15], respectively. Furthermore, assume $|\mathcal{F}_\lambda| \leq \mathrm{poly}(\lambda)$ and $|\mathcal{M}_\lambda| \leq \mathrm{poly}(\lambda)$. Then the* RPE *constructed above is function hiding (Def. 2.6).*

Function $\mathsf{Re\text{-}Enc}[\mathsf{lb}, x, \gamma_1, \gamma_2, \delta]$

Hardwired values: A label lb, an attribute x, and SKE ciphertexts γ_1, γ_2, and δ.

Inputs: A kpABE master public key $\mathsf{kpABE.mpk}$, a function $f \in \mathcal{F}_\lambda$, a message $m \in \mathcal{M}_\lambda$, a PRF key K, a trapdoor mode $\mathsf{mode} \in \{0, 1, 2\}$ and SKE keys $\mathsf{SKE.key}_1$ and $\mathsf{SKE.key}_2$.

Output : A kpABE ciphertext.

1. Parse the input as $(\mathsf{ABE.mpk}, f, m, K, \mathsf{mode}, \mathsf{SKE.key}_1, \mathsf{SKE.key}_2)$.
2. Set $\tilde{m} = \begin{cases} m & \text{if } f(x) = 1 \\ 0 & \text{if } f(x) = 0. \end{cases}$
3. Compute $\mathsf{kpABE.ct}_{\mathsf{lb}} = \mathsf{kpABE.Enc}(\mathsf{kpABE.mpk}, \mathsf{lb}, \tilde{m}; F(K, (\mathsf{lb}, x)))$.
4. Compute $\mathsf{flag} = \mathsf{SKE.Dec}(\mathsf{SKE.key}_2, \delta)$.
5. Compute $\mathsf{out}_i = \mathsf{SKE.Dec}(\mathsf{SKE.key}_i, \gamma_i)$ for $i \in \{1, 2\}$.
6. If $\mathsf{mode} = 0$, output $\mathsf{kpABE.ct}_{\mathsf{lb}}$.
7. If $\mathsf{mode} = 1$, output out_1.
8. If $\mathsf{mode} = 2$, output $\begin{cases} \mathsf{out}_2 & \text{if } \mathsf{flag} = 1 \\ \mathsf{kpABE.ct}_{\mathsf{lb}} & \text{if } \mathsf{flag} = 0. \end{cases}$

Fig. 2. Function to compute kpABE ciphertexts depending on various conditions.

Theorem 3.3. *Assume that F is a secure* PRF, SKE *is correct and secure,* FE *and* kpABE *are secure as per Definitions [2, Def. 2.6] and [2, Def. 2.15], respectively. Furthermore, assume $|\mathcal{F}_\lambda| \leq \mathrm{poly}(\lambda)$ and $|\mathcal{M}_\lambda| \leq \mathrm{poly}(\lambda)$. Then the construction for* RPE *satisfies message hiding property as defined in Def. 2.3.*

Due to space constraints, we provide the proofs in the full version.

Efficiency. Here we argue that our construction achieves optimal parameters. Namely, we show that the size of each parameter is independent of $|L|$. For details, please see the full version.

3.1 Alternate Construction using LOT

The construction is similar to the above except that we use LOT in place of ABE, which brings in the following changes in the KeyGen, Enc, and Dec algorithms:

- We use $\mathsf{LOT} = (\mathsf{LOT.crsGen}, \mathsf{LOT.Hash}, \mathsf{LOT.Send}, \mathsf{LOT.Receive})$ instead of kpABE.
- The function in Figure 2, for which FE key is generated now takes as input LOT objects crs and digest, instead of $\mathsf{kpABE.mpk}$ and computes $\mathsf{LOT.ct}_{\mathsf{lb}} = \mathsf{LOT.Send}(\mathsf{crs}, \mathsf{digest}, \mathsf{lb}, 0, \tilde{m}; F(K, (\mathsf{lb}, x)))$, instead of $\mathsf{kpABE.ct}_{\mathsf{lb}}$.

– The encryption algorithm changes as follows:
$\mathsf{RPE.Enc}(\mathsf{RPE.mpk}, f, m, L) \to \mathsf{RPE.ct}$. The encryption algorithm does the following:
  - Parse $\mathsf{RPE.mpk} = \mathsf{FE.mpk}$ and sample a PRF key $K \leftarrow \{0,1\}^\lambda$.
  - Generate $\mathsf{crs} \leftarrow \mathsf{LOT.crsGen}(1^\lambda)$.
  - Compute $(\mathsf{digest}, \hat{D}) \leftarrow \mathsf{LOT.Hash}(\mathsf{crs}, D)$, where D is a binary vector of length N (the no. of users) and is 1 at positions corresponding to non-revoked labels, i.e. $D[\mathsf{lb}'] = 1$ iff $\mathsf{lb}' \notin L$.
  - Compute $\mathsf{FE.ct} \leftarrow \mathsf{FE.Enc}(\mathsf{FE.mpk}, (\mathsf{crs}, \mathsf{digest}, f, m, K, 0, \bot, \bot))$.
  - Output $\mathsf{RPE.ct} = (\mathsf{crs}, \mathsf{FE.ct})$.
– The algorithm for decryption also changes accordingly as follows:
$\mathsf{RPE.Dec}(\mathsf{RPE.sk}_{\mathsf{lb},x}, \mathsf{RPE.ct}, L) \to m'$. The decryption algorithm does the following:
  - Parse $\mathsf{RPE.ct} = (\mathsf{crs}, \mathsf{FE.ct})$ and $\mathsf{RPE.sk}_{\mathsf{lb},x} = \mathsf{FE.sk}_{\mathsf{lb},x}$.
  - Define D from L as described in the encryption algorithm and compute $(\mathsf{digest}, \hat{D}) \leftarrow \mathsf{LOT.Hash}(\mathsf{crs}, D)$.
  - Compute $\mathsf{LOT.ct}' = \mathsf{FE.Dec}(\mathsf{FE.sk}_{\mathsf{lb},x}, \mathsf{FE.ct})$.
  - Compute $m' = \mathsf{LOT.Receive}^{\hat{D}}(\mathsf{crs}, \mathsf{LOT.ct}', \mathsf{lb})$.
  - Output m'.

We provide the proofs for correctness and security in the full version.

4 Revocable Mixed Functional Encryption

4.1 Definition

A revocable mixed functional encryption (RMFE) scheme with input domain $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in[\mathbb{N}]}$, a function family $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in[\mathbb{N}]}$ where $\mathcal{F}_\lambda = \{f : \mathcal{X}_\lambda \to \{0,1\}\}$, a label space $\mathcal{L} = \{\mathcal{L}_\lambda\}_{\lambda\in[\mathbb{N}]}$ has the following syntax.

$\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$. The setup algorithm takes as input the security parameter λ and outputs a master public key mpk and a master secret key msk.

$\mathsf{KeyGen}(\mathsf{msk}, \mathsf{lb}, x) \to \mathsf{sk}_{\mathsf{lb},x}$. The key generation algorithm takes as input the master secret key msk, a label $\mathsf{lb} \in \mathcal{L}_\lambda$ and an input $x \in \mathcal{X}_\lambda$. It outputs a secret key $\mathsf{sk}_{\mathsf{lb},x}$.

$\mathsf{PK\text{-}Enc}(\mathsf{mpk}, L) \to \mathsf{ct}$. The public key encryption algorithm takes as input the master public key mpk and a revocation list $L \subseteq \mathcal{L}_\lambda$ and outputs a ciphertext ct.

$\mathsf{SK\text{-}Enc}(\mathsf{msk}, f, L) \to \mathsf{ct}$. The secret key encryption algorithm takes as input the master secret key msk, a function $f \in \mathcal{F}_\lambda$ and a revocation list $L \subseteq \mathcal{L}_\lambda$, and outputs a ciphertext ct.

$\mathsf{Dec}(\mathsf{sk}_{\mathsf{lb},x}, L, \mathsf{ct}) \to \{0,1\}$. The decryption algorithm takes the secret key $\mathsf{sk}_{\mathsf{lb},x}$, a revocation list $L \subseteq \mathcal{L}_\lambda$ and a ciphertext ct and outputs a bit.

Definition 4.1 (Correctness). *A* RMFE *scheme is said to be correct if there exists negligible functions* $\mathsf{negl}_1(\cdot), \mathsf{negl}_2(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*, the following holds*

$$\mathsf{Pr}\left[\mathsf{Dec}(\mathsf{sk}_{\mathsf{lb},x}, L, \mathsf{ct}) = 1 : \begin{array}{c}(\mathsf{mpk},\mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda);\\ \mathsf{sk}_{\mathsf{lb},x} \leftarrow \mathsf{KeyGen}(\mathsf{msk},\mathsf{lb},x);\\ \mathsf{ct} \leftarrow \mathsf{PK\text{-}Enc}(\mathsf{mpk}, L)\end{array}\right] \geq 1 - \mathsf{negl}_1(\lambda).$$

$$\mathsf{lb} \notin L \Rightarrow \mathsf{Pr}\left[\mathsf{Dec}(\mathsf{sk}_{\mathsf{lb},x}, L, \mathsf{ct}) = f(x) : \begin{array}{c}(\mathsf{mpk},\mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda);\\ \mathsf{sk}_{\mathsf{lb},x} \leftarrow \mathsf{KeyGen}(\mathsf{msk},\mathsf{lb},x);\\ \mathsf{ct} \leftarrow \mathsf{SK\text{-}Enc}(\mathsf{msk}, f, L)\end{array}\right] \geq 1-\mathsf{negl}_2(\lambda).$$

Security. Here we define the security requirements of RMFE scheme.

Definition 4.2 (q-query Mode Hiding). *Let* $q(\cdot)$ *be any fixed polynomial. A* RMFE *scheme satisfies* q*-query mode hiding security if for every PPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for every* $\lambda \in \mathbb{N}$*,*

$$\mathsf{Pr}\left[\beta' = \beta : \begin{array}{l}(\mathsf{mpk},\mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda);\\ f, L \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot),\mathsf{SK\text{-}Enc}(\mathsf{msk},\cdot,\cdot)}(\mathsf{mpk});\\ \beta \leftarrow \{0,1\}; \mathsf{ct}_0 \leftarrow \mathsf{SK\text{-}Enc}(\mathsf{msk}, f, L);\\ \mathsf{ct}_1 \leftarrow \mathsf{PK\text{-}Enc}(\mathsf{mpk}, L);\\ \beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot),\mathsf{SK\text{-}Enc}(\mathsf{msk},\cdot,\cdot)}(\mathsf{ct}_\beta)\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathcal{A}$ *can make at most* $q(\lambda)$ *queries to the* $\mathsf{SK\text{-}Enc}(\mathsf{msk},\cdot,\cdot)$ *oracle and is admissible only if for all the key queries* (lb, x) *to the* $\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot)$ *oracle,* $f(x) = 1$*.*

Definition 4.3 (q-query Selective Function Hiding). *Let* $q(\cdot)$ *be any fixed polynomial. A* RMFE *scheme satisfies* q*-query selective function hiding security if for every PPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for every* $\lambda \in \mathbb{N}$*,*

$$\mathsf{Pr}\left[\beta' = \beta : \begin{array}{l}L \leftarrow \mathcal{A}(1^\lambda);\\ (\mathsf{mpk},\mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda);\\ (f_0, f_1) \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot),\mathsf{SK\text{-}Enc}(\mathsf{msk},\cdot,\cdot)}(\mathsf{mpk});\\ \beta \leftarrow \{0,1\}; \mathsf{ct}_\beta \leftarrow \mathsf{SK\text{-}Enc}(\mathsf{msk}, f_\beta, L);\\ \beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot),\mathsf{SK\text{-}Enc}(\mathsf{msk},\cdot,\cdot)}(\mathsf{ct}_\beta)\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

where $\mathcal{A}$ *can make at most* $q(\lambda)$ *queries to the* $\mathsf{SK\text{-}Enc}(\mathsf{msk},\cdot,\cdot)$ *oracle and for all the key queries* (lb, x) *to the* $\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot)$ *oracle, either* $f_0(x) = f_1(x)$ *or* $\mathsf{lb} \in L$*.*

Remark 4.4. We note that when the function space $\mathcal{F}_\lambda$ is polynomially small and q is a constant, a variant of Definition 4.3 where the adversary outputs the challenge functions (f_0, f_1) and the SK-Enc query functions $\{\bar{f}_i\}_{i\in[q]}$ at the beginning of the game, before the $\mathsf{Setup}(1^\lambda)$ algorithm is run, is equivalent to Definition 4.3 where the adversary adaptively outputs the challenge functions (f_0, f_1) and can make SK-Enc queries adaptively, with polynomial loss. Similar comment also applies to Definition 4.2. We will use these simplifications in the security proofs.

4.2 Construction

In this section we give a construction of 1-query secure RMFE scheme, with input space $\mathcal{X} = \{\mathcal{X}_\lambda\}_\lambda$, a function family $\mathcal{F} = \{\mathcal{F}_\lambda\}_\lambda$ where $\mathcal{F}_\lambda = \{f : \mathcal{X}_\lambda \to \{0,1\}\}$ and a label space $\mathcal{L} = \{\mathcal{L}_\lambda\}_\lambda$. We assume that the size of $|\mathcal{F}_\lambda|$ is bounded by some polynomial in λ, which will suffice for our purpose.
Our scheme uses the following building blocks:

1. A 2-bounded semi-adaptive simulation based function-message private ([2, Def. 2.11]) SKFE scheme $\mathsf{SKFE} = (\mathsf{SKFE.Setup}, \mathsf{SKFE.KeyGen}, \mathsf{SKFE.Enc}, \mathsf{SKFE.Dec})$ that supports the function class $\mathcal{F}$. This can be instantiated from one-way functions ([2, Lemma 2.12]).
2. A key-policy ABE scheme $\mathsf{kpABE} = (\mathsf{kpABE.Setup}, \mathsf{kpABE.Enc}, \mathsf{kpABE.KeyGen}, \mathsf{kpABE.Dec})$ for the circuit class $\mathcal{C}_{\ell(\lambda),d(\lambda)}$ with message space $\{0,1\}^\lambda$ satisfying Sel-IND security ([2, Def. 2.14]) and efficiency properties described in [2, Theorem 2.18]. We set $\ell(\lambda) = \ell_{\mathsf{lb}} + \log(\lambda) + 1$ and $d(\lambda) = \omega(\log \lambda)$, where ℓ_{lb} is the label length.[11] This can be instantiated from the LWE assumption ([2, Theorem 2.18]).
3. A lockable obfuscation scheme $\mathsf{LO} = (\mathsf{LO.Obf}, \mathsf{LO.Eval})$ with lock space $\{0,1\}^\lambda$ that supports circuits of the form CC defined in Fig. 3. As we will analyze later, the circuit is of fixed polynomial size in λ and $|f|$, where $|f|$ is the description size of the function $f \in \mathcal{F}$. This can be instantiated from the LWE assumption ([2, Theorem 2.25].).

Below we describe our construction of a 1-query secure RMFE scheme $\mathsf{RMFE} = (\mathsf{RMFE.Setup}, \mathsf{RMFE.KeyGen}, \mathsf{RMFE.PK\text{-}Enc}, \mathsf{RMFE.SK\text{-}Enc}, \mathsf{RMFE.Dec})$.

$\mathsf{RMFE.Setup}(1^\lambda) \to (\mathsf{RMFE.mpk}, \mathsf{RMFE.msk})$. The setup algorithm does the following:
- Generate $\mathsf{SKFE.msk} \leftarrow \mathsf{SKFE.Setup}(1^\lambda)$.
- Generate $(\mathsf{kpABE.mpk}, \mathsf{kpABE.msk}) \leftarrow \mathsf{kpABE.Setup}(1^\lambda)$.
- Output $\mathsf{RMFE.mpk} = \mathsf{kpABE.mpk}$ and $\mathsf{RMFE.msk} = (\mathsf{SKFE.msk}, \mathsf{kpABE.mpk}, \mathsf{kpABE.msk})$.

$\mathsf{RMFE.KeyGen}(\mathsf{RMFE.msk}, \mathsf{lb}, x) \to \mathsf{RMFE.sk}_{\mathsf{lb},x}$. The key generation algorithm does the following:
- Parse $\mathsf{RMFE.msk} = (\mathsf{SKFE.msk}, \mathsf{kpABE.mpk}, \mathsf{kpABE.msk})$.
- For all $j \in [\lambda], b \in \{0,1\}$, sample $K_{j,b}, R_{j,b} \leftarrow \{0,1\}^\lambda$. Denote $K = \{K_{j,b}\}_{j\in[\lambda],b\in\{0,1\}}$ and $R = \{R_{j,b}\}_{j\in[\lambda],b\in\{0,1\}}$.
- Compute
$$\mathsf{SKFE.ct} \leftarrow \mathsf{SKFE.Enc}(\mathsf{SKFE.msk}, (x, K, R)).$$
- For all $j \in [\lambda], b \in \{0,1\}$, compute
$$\mathsf{kpABE.ct}_{\mathsf{lb},j,b} \leftarrow \mathsf{kpABE.Enc}(\mathsf{kpABE.mpk}, (\mathsf{lb}, j, b), K_{j,b}).$$

[11] Concretely, we can choose $d(\lambda) = \Theta(\log \lambda \log\log \lambda)$ for example.

– Output $\mathsf{RMFE.sk}_{\mathsf{lb},x} = (\mathsf{SKFE.ct}, \mathsf{kpABE.mpk}, \{(\mathsf{lb}, j, b), \mathsf{kpABE.ct}_{\mathsf{lb},j,b}\}_{j\in[\lambda],b\in\{0,1\}})$.

$\mathsf{RMFE.PK\text{-}Enc}(\mathsf{RMFE.mpk}, L) \to \mathsf{RMFE.ct}$. The public key encryption algorithm does the following:

– Computes a simulated code $\mathsf{RMFE.ct} \leftarrow \mathsf{LO.Sim}(1^\lambda, 1^{|\mathsf{CC}|})$[12].

– It outputs RMFE.ct as the ciphertext.

$\mathsf{RMFE.SK\text{-}Enc}(\mathsf{RMFE.msk}, f, L) \to \mathsf{RMFE.ct}$. The secret key encryption algorithm does the following:

– Parse $\mathsf{RMFE.msk} = (\mathsf{SKFE.msk}, \mathsf{kpABE.mpk}, \mathsf{kpABE.msk})$, and sample a tag $\mathbf{z} \leftarrow \{0,1\}^\lambda$ and a lock value $\alpha \leftarrow \{0,1\}^\lambda$.

– For all $j \in [\lambda]$, compute $\mathsf{kpABE.sk}_{L,j,z_j} \leftarrow \mathsf{kpABE.KeyGen}(\mathsf{kpABE.msk}, C_{L,j,z_j})$, where the function C_{L,j,z_j} has L, j and z_j hardwired and is defined as follows :
On input $(\mathsf{lb}, i, b) \in \mathcal{L}_\lambda \times [\lambda] \times \{0,1\}$,

$$C_{L,j,z_j}(\mathsf{lb}, i, b) = \begin{cases} 1 & \text{if } (\mathsf{lb} \notin L) \wedge (i = j) \wedge (b = z_j) \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

– Compute $\mathsf{SKFE.sk} \leftarrow \mathsf{SKFE.KeyGen}(\mathsf{SKFE.msk}, P_{f,\mathbf{z},\alpha})$, where the function $P_{f,\mathbf{z},\alpha}$ has $f, \mathbf{z}, \alpha$ hardwired and is defined as follows :
On input $x \in \mathcal{X}_\lambda$, $K = \{K_{j,b}\}_{j\in[\lambda],b\in\{0,1\}}$, $R = \{R_{j,b}\}_{j\in[\lambda],b\in\{0,1\}}$,

$$P_{f,\mathbf{z},\alpha}(x, K, R) = \begin{cases} \bigoplus_j K_{j,z_j} \oplus \alpha & \text{if } f(x) = 0 \\ \bigoplus_j R_{j,z_j} & \text{if } f(x) = 1. \end{cases} \tag{4.2}$$

– Construct function $\mathsf{CC}[\mathsf{SKFE.sk}, \{\mathsf{kpABE.sk}_{L,j,z_j}\}_{j\in[\lambda]}]$, with SKFE.sk and $\{\mathsf{kpABE.sk}_{L,j,z_j}\}_{j\in[\lambda]}$ hardwired and is defined as in Figure 3.

– Output $\mathsf{RMFE.ct} \leftarrow \mathsf{LO.Obf}(\mathsf{CC}[\mathsf{SKFE.sk}, \{\mathsf{kpABE.sk}_{L,j,z_j}\}_{j\in[\lambda]}], \alpha)$.

$\mathsf{RMFE.Dec}(\mathsf{RMFE.sk}_{\mathsf{lb},x}, \mathsf{RMFE.ct}, L) \to \{0,1\}$. The decryption algorithm does the following:

– Parse $\mathsf{RMFE.sk}_{\mathsf{lb},x} = (\mathsf{SKFE.ct}, \mathsf{kpABE.mpk}, \{(\mathsf{lb}, j, b), \mathsf{kpABE.ct}_{\mathsf{lb},j,b}\}_{j\in[\lambda],b\in\{0,1\}})$ and $\mathsf{RMFE.ct} = \widetilde{\mathsf{CC}}$, where $\widetilde{\mathsf{CC}}$ is regarded as an obfuscated circuit of LO.

– For all $j \in [\lambda], b \in \{0,1\}$, compute

$$\mathsf{kpABE.off}_{\mathsf{lb},j,b} \leftarrow \mathsf{kpABE.Dec}^{\mathsf{off}}(\mathsf{kpABE.mpk}, C_{L,j,b}, (\mathsf{lb}, j, b)).$$

– Compute

$$y = \mathsf{LO.Eval}\big(\widetilde{\mathsf{CC}}, (\mathsf{SKFE.ct}, \{\mathsf{kpABE.ct}_{\mathsf{lb},j,b}, \mathsf{kpABE.off}_{\mathsf{lb},j,b}\}_{j\in[\lambda],b\in\{0,1\}})\big).$$

– Output 1 if $y = \perp$, else output 0.

Remark 4.5. We note that by performing the part of the ABE decryption that uses $C_{L,j,b}$, outside of CC , we do not need to provide $C_{L,j,b}$ (or L) as input to CC. Instead, we provide $\{\mathsf{kpABE.off}_{\mathsf{lb},j,b}\}_{j\in[\lambda],b\in\{0,1\}}$ whose size is independent of the size of $C_{L,j,b}$ (and thus that of L). This helps us in getting succinct ciphertext.

[12] Here, CC represents the maximum possible size of $\mathsf{CC}[\cdot,\cdot]$ circuit defined in Figure 3.

Function $\mathsf{CC}[\mathsf{SKFE.sk}, \{\mathsf{kpABE.sk}_{L,j,z_j}\}_{j\in[\lambda]}]$

Hardwired values: A SKFE secret key SKFE.sk and kpABE keys $\{\mathsf{kpABE.sk}_{L,j,z_j}\}_{j\in[\lambda]}$.

Inputs: A SKFE ciphertext SKFE.ct and kpABE ciphertexts $\{\mathsf{kpABE.ct}_{\mathsf{lb},j,b}, \mathsf{kpABE.off}_{\mathsf{lb},j,b}\}_{j\in[\lambda],b\in\{0,1\}}$.

Output : A binary string $\alpha^* \in \{0,1\}^\lambda$.

1. For all $j \in [\lambda]$, compute $m_j = \mathsf{kpABE.Dec}^{\mathsf{on}}(\mathsf{kpABE.sk}_{L,j,z_j}, \mathsf{kpABE.ct}_{\mathsf{lb},j,z_j}, \mathsf{kpABE.off}_{\mathsf{lb},j,z_j})$.
   Let $M_0 = \bigoplus_j m_j$
2. Compute $M_1 = \mathsf{SKFE.Dec}(\mathsf{SKFE.sk}, \mathsf{SKFE.ct})$.
3. Output $M_1 \oplus M_0$.

Fig. 3. Compute and Compare function CC

Correctness and Security. We prove the correctness and security via the following theorems.

Theorem 4.6. *Suppose* kpABE, LO *and* SKFE *are correct and* LO *is secure, then the above construction of* RMFE *satisfies correctness as defined in Definition 4.1.*

Theorem 4.7. *Assume that* SKFE *and* LO *are secure as per Definitions [2, Def. 2.11] and [2, Def. 2.24], respectively. Furthermore, assume* $|\mathcal{F}_\lambda| \leq \mathrm{poly}(\lambda)$. *Then the* RMFE *construction satisfies* 1*-query mode hiding security as per Definition 4.2.*

Theorem 4.8. *Assume* SKFE *is secure ([2, Def. 2.11]),* kpABE *satisfies* Sel-IND *security ([2, Def. 2.14]). Furthermore, assume* $|\mathcal{F}_\lambda| \leq \mathrm{poly}(\lambda)$. *Then, the* RMFE *construction satisfies* 1*-query function hiding as defined in Definition 4.3.*

Due to space constraints, we prove these theorems in the full version.

Efficiency. Here we argue that our construction achieves optimal parameters. Namely, we show that the sizes of the parameters are independent of $|L|$. For details, please see the full version.

Acknowledgements. We thank the reviewers of Eurocrypt 2023 for helpful comments, especially for suggesting the alternative construction of RPE based on FE and laconic OT. This work was supported in part by the DST "Swarnajayanti" fellowship, Cybersecurity Center of Excellence, IIT Madras, National Blockchain Project and the Algorand Centres of Excellence programme managed by Algorand Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of sponsors. The fourth author was partially supported by JST AIP Acceleration Research JPMJCR22U5 and JSPS KAKENHI Grant Number 19H01109, Japan.

References

1. Agrawal, S.: Indistinguishability obfuscation without multilinear maps: New techniques for bootstrapping and instantiation. In: Eurocrypt (2019). https://doi.org/10.1007/978-3-030-17653-2_7
2. Agrawal, S., Kumari, S., Yadav, A., Yamada, S.: Trace and revoke with optimal parameters from polynomial hardness. Cryptology ePrint Archive (2022), https://eprint.iacr.org/2022/1347.pdf
3. Agrawal, S., Maitra, M.: Fe and io for turing machines from minimal assumptions. In: TCC. Springer (2018). https://doi.org/10.1007/978-3-030-03810-6_18
4. Agrawal, S., Pellet-Mary, A.: Indistinguishability obfuscation without maps: Attacks and fixes for noisy linear fe. In: Eurocrypt (2020)
5. Agrawal, S., Wichs, D., Yamada, S.: Optimal broadcast encryption from lwe and pairings in the standard model. In: TCC (2020). https://doi.org/10.1007/978-3-030-64375-1_6
6. Agrawal, S., Yamada, S.: Optimal broadcast encryption from pairings and lwe. In: EUROCRYPT (2020). https://doi.org/10.1007/978-3-030-45721-1_2
7. Ananth, P., Brakerski, Z., Segev, G., Vaikuntanathan, V.: From selective to adaptive security in functional encryption. In: CRYPTO (2015). https://doi.org/10.1007/978-3-662-48000-7_32
8. Ananth, P., Jain, A.: Indistinguishability obfuscation from compact functional encryption. In: CRYPTO (2015). https://doi.org/10.1007/978-3-662-47989-6_15
9. Attrapadung, N., Libert, B., Panafieu, E.d.: Expressive key-policy attribute-based encryption with constant-size ciphertexts. In: PKC. Springer (2011). https://doi.org/10.1007/978-3-642-19379-8_6
10. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S., Yang, K.: On the (im)possibility of obfuscating programs. In: CRYPTO (2001). https://doi.org/10.1007/3-540-44647-8_1
11. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy. pp. 321–334 (2007). https://doi.org/10.1109/SP.2007.11
12. Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation from functional encryption. FOCS (2015). https://doi.org/10.1109/FOCS.2015.20, http://eprint.iacr.org/2015/163
13. Boneh, D., Gentry, C., Gorbunov, S., Halevi, S., Nikolaenko, V., Segev, G., Vaikuntanathan, V., Vinayagamurthy, D.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: EUROCRYPT (2014). https://doi.org/10.1007/978-3-642-55220-5_30
14. Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: CRYPTO (2005). https://doi.org/10.1007/11535218_16
15. Boneh, D., Waters, B., Zhandry, M.: Low overhead broadcast encryption from multilinear maps. In: CRYPTO (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. Algorithmica **79**(4), 1233–1285 (2017). https://doi.org/10.1007/978-3-662-44371-2_27
17. Boyen, X., Waters, B.: Anonymous hierarchical identity-based encryption (without random oracles). In: CRYPTO (2006). https://doi.org/10.1007/11818175_17

18. Brakerski, Z., Komargodski, I., Segev, G.: Multi-input functional encryption in the private-key setting: Stronger security from weaker assumptions. In: EUROCRYPT. Springer (2016). https://doi.org/10.1007/978-3-662-49896-5_30
19. Brakerski, Z., Segev, G.: Function-private functional encryption in the private-key setting. Journal of Cryptology **31**(1), 202–225 (2018). https://doi.org/10.1007/s00145-017-9255-y
20. Caro, A.D., Iovino, V., Jain, A., O'Neill, A., Paneth, O., Persiano, G.: On the achievability of simulation-based security for functional encryption. In: CRYPTO (2013). https://doi.org/10.1007/978-3-642-40084-1_29
21. Chen, Y., Vaikuntanathan, V., Waters, B., Wee, H., Wichs, D.: Traitor-tracing from lwe made simple and attribute-based. In: TCC (2018). https://doi.org/10.1007/978-3-030-03810-6_13
22. Cho, C., Döttling, N., Garg, S., Gupta, D., Miao, P., Polychroniadou, A.: Laconic oblivious transfer and its applications. In: CRYPTO (2017). https://doi.org/10.1007/978-3-319-63715-0_2
23. Chor, B., Fiat, A., Naor, M.: Tracing traitors. In: CRYPTO (1994). https://doi.org/10.1007/3-540-48658-5_25
24. Coron, J., Lepoint, T., Tibouchi, M.: Practical multilinear maps over the integers. In: CRYPTO (2013). https://doi.org/10.1007/978-3-642-40041-4_26
25. Devadas, L., Quach, W., Vaikuntanathan, V., Wee, H., Wichs, D.: Succinct lwe sampling, random polynomials, and obfuscation. In: TCC (2021). https://doi.org/10.1007/978-3-030-90453-1_9
26. Fiat, A., Naor, M.: Broadcast encryption. In: Stinson, D.R. (ed.) CRYPTO (1993). https://doi.org/10.1007/3-540-48329-2_40
27. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: EUROCRYPT (2013). https://doi.org/10.1007/978-3-642-38348-9_1
28. Garg, S., Pandey, O., Srinivasan, A.: Revisiting the cryptographic hardness of finding a nash equilibrium. In: CRYPTO (2016). https://doi.org/10.1007/978-3-662-53008-5_20
29. Garg, S., Pandey, O., Srinivasan, A., Zhandry, M.: Breaking the sub-exponential barrier in obfustopia. In: EUROCRYPT (2017). https://doi.org/10.1007/978-3-319-56617-7_6
30. Gay, R., Pass, R.: Indistinguishability obfuscation from circular security. In: STOC (2021). https://doi.org/10.1145/3406325.3451070
31. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute based encryption for circuits. In: STOC (2013). https://doi.org/10.1145/2488608.2488677
32. Goyal, R., Koppula, V., Waters, B.: Lockable obfuscation. In: FOCS (2017). https://doi.org/10.1109/FOCS.2017.62
33. Goyal, R., Koppula, V., Waters, B.: Collusion resistant traitor tracing from learning with errors. In: STOC (2018). https://doi.org/10.1145/3188745.3188844
34. Goyal, R., Koppula, V., Waters, B.: New approaches to traitor tracing with embedded identities. In: TCC (2019). https://doi.org/10.1007/978-3-030-36033-7_6
35. Goyal, R., Quach, W., Waters, B., Wichs, D.: Broadcast and trace with n^{ϵ} ciphertext size from standard assumptions. In: Crypto (2019), https://eprint.iacr.org/2019/636
36. Goyal, R., Vusirikala, S., Waters, B.: Collusion resistant broadcast and trace from positional witness encryption. In: PKC (2019). https://doi.org/10.1007/978-3-030-17259-6_1
37. Jain, A., Lin, H., Lou, P., Sahai, A.: Polynomial-time cryptanalysis of the subspace flooding assumption for post-quantum io. In: EUROCRYPT (2023)

38. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from well-founded assumptions. In: STOC (2021). https://doi.org/10.1145/3406325.3451093
39. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from lpn over large fields, dlin, and constant depth prgs. In: EUROCRYPT (2022). https://doi.org/10.1007/978-3-031-06944-4_23
40. Kim, S., Wu, D.J.: Collusion resistant trace-and-revoke for arbitrary identities from standard assumptions. In: ASIACRYPT (2020). https://doi.org/10.1007/978-3-030-64834-3_3
41. Komargodski, I., Segev, G.: From minicrypt to obfustopia via private-key functional encryption. Journal of Cryptology **33**(2), 406–458 (2020). https://doi.org/10.1007/978-3-319-56620-7_5
42. Lin, H., Pass, R., Seth, K., Telang, S.: Output-compressing randomized encodings and applications. In: TCC (2016). https://doi.org/10.1007/978-3-662-49096-9_5
43. Naor, M., Pinkas, B.: Efficient trace and revoke schemes. International Journal of Information Security **9**(6), 411–424 (2010). https://doi.org/10.1007/s10207-010-0121-2
44. Nishimaki, R., Wichs, D., Zhandry, M.: Anonymous traitor tracing: how to embed arbitrary information in a key. In: EUROCRYPT (2016). https://doi.org/10.1007/978-3-662-49896-5_14
45. Takashima, K.: Expressive attribute-based encryption with constant-size ciphertexts from the decisional linear assumption. In: SCN (2014). https://doi.org/10.1007/978-3-319-10879-7_17
46. Tsabary, R.: Candidate witness encryption from lattice techniques. In: Crypto (2022)
47. Vaikuntanathan, V., Wee, H., Wichs, D.: Witness encryption and null-io from evasive LWE. In: Asiacrypt (2022)
48. Wee, H.: Optimal broadcast encryption and cp-abe from evasive lattice assumptions. In: Eurocrypt (2022). https://doi.org/10.1007/978-3-031-07085-3_8
49. Wee, H., Wichs, D.: Candidate obfuscation via oblivious lwe sampling. In: EUROCRYPT. Springer (2021). https://doi.org/10.1007/978-3-030-77883-5_5
50. Wichs, D., Zirdelis, G.: Obfuscating compute-and-compare programs under LWE. In: FOCS (2017). https://doi.org/10.1109/FOCS.2017.61
51. Zhandry, M.: New techniques for traitor tracing: Size $N^{1/3}$ and more from pairings. In: CRYPTO (2020). https://doi.org/10.1007/978-3-030-56784-2_22